FACULTY OF MATHEMATICS AND PHYSICS, CHARLES UNIVERSITY

DIPLOMA THESIS

A Complete DLT-based Camera Calibration with a Virtual 3D Calibration Object

Author: Hynek Bakstein
Specialization: Computer science

Supervisor: Mgr. Radim Halíř
Faculty of Mathematics and Physics, Charles University, Prague

Prague 1999

Contents

Preface **iii**

1 Introduction **1**

2 Geometrical Calibration of a Camera **5**

2.1 Calibration Objects 5

2.2 Control Points 7

2.3 Models of a Camera 10

2.3.1 Pinhole Model 10

2.3.2 Direct Linear Transform (DLT) Model 12

2.3.3 Coplanar DLT Model 12

2.3.4 Extended DLT Models 13

2.3.5 Traditional Photogrammetry Model 13

2.3.6 Simplified Traditional Models 14

2.3.7 Two-plane Camera Models 14

2.3.8 Comparison of Camera Models 14

3 Camera Calibration with a Virtual 3D Calibration Object **16**

3.1 Motivation 16

3.2 Overview of the Proposed Calibration Method 17

3.3 Step 1 – Initial Estimate of the Intrinsic Camera Parameters . 19

3.4 Step 2 – Estimation of the Extrinsic Camera Parameters for Each View 19

3.4.1 Direct Computation of the Extrinsic Parameters 20
3.4.2 Finding the Optimal Extrinsic Parameters 23
3.4.3 Reduction of the Dimension of the Searching Space . . 24
3.5 Step 3 – Construction of a Virtual 3D Calibration Object . . . 25
3.6 Step 4 – Calibration with the Virtual 3D Calibration Object . 28
3.7 Iterative Refinement of the Camera Parameters 28
3.8 Discussion 29

4 Practical Realization 30

5 Experimental Results 32
5.1 Evaluation on Synthetic Data 32
5.2 Tests with Real Data 38
5.3 Comparison with Other Calibration Methods 41
5.4 Discussion 44

6 Conclusion and Outlook 45

A Description of the Calibration Toolbox 47

B Toolbox User Instructions 48

C Example of the Calibration Process 50

Bibliography 54

Preface

Summary

This work is devoted to a new camera calibration method. The method is based on a simulation of a virtual 3D object by multiple views of a coplanar calibration grid. The method refines values of the estimated camera parameters through an iterative process consisting of several steps. The approach has been evaluated in many experiments with both synthetic and real calibration targets.

Structure of This Work

Chapter 1 gives a brief introduction to a camera calibration. Different camera models and calibration approaches are discussed in Chapter 2. A new calibration method is proposed in Chapter 3 which is the main part of this work. A practical realization of the proposed method is presented in Chapter 4. Chapter 5 is devoted to an experimental evaluation of the method. Chapter 6 concludes the whole work and gives some future directions. The appropriate calibration toolbox created by the author is described in Appendix A. Appendix B gives the user instructions for this toolbox. Finally, an example of the calibration process is listed in Appendix C.

Contributions of This Work

The main contributions of this work are:

- A new iterative camera calibration method using multiple views of coplanar calibration target (Chapter 3)

- A MATLAB calibration toolbox implementing the proposed method (Appendix A)
- A througout overview of geometrical calibration of a camera (Chapter 2)
- A new approach to 3 by 3 coplanar DLT matrix decomposition based on non-linear minimization (Section 3.4)
- One parameter expression of the rotation in the optimization part (Section 3.4.3)

Basic parts of this work were covered by the paper "Bakstein, Halíř: Camera Calibration with a Simulated Three Dimensional Calibration Object" which was accepted for the 23-rd Workshop of the Austrian Association for Pattern Recognition (ÖAGM'99).

Acknowledgements

I would like to thank my diploma thesis supervisor Radim Halíř. Without his aid and knowledge I would have hardly finished this work. My thanks also pertain to my parents for their patience and support.

Chapter 1

Introduction

Every image acquired by a camera is distorted. The distortion is caused by the lenses, improper manufacturing and positioning of the camera sensor and even by temperature and vibrations [Nal93]. Figure 1.1 depicts forming of an image of a scene. The scene is observed by a camera. Due to a lower cost, an analogue camera is typically used which requires a frame grabber to convert an analogue signal into an appropriate digital form. The *line jitter* distortion caused by different sampling frequencies in the camera and the frame grabber is introduced during the conversion. After the digitalization, the image is ready to be displayed and visualized by a computer.

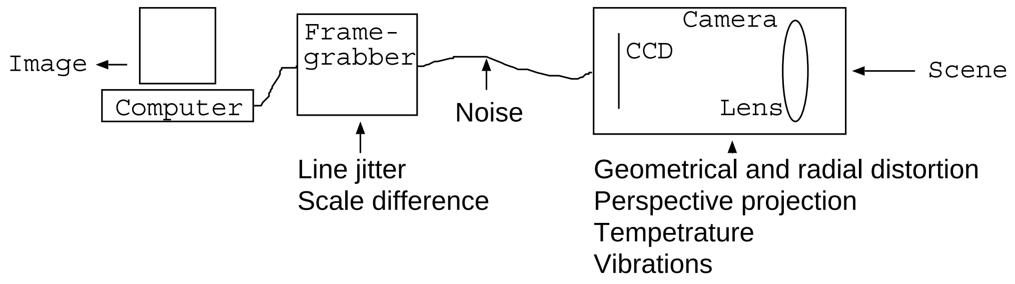


Figure 1.1: A scheme of an image forming. See text for a description

A mapping of a 3D scene into a 2D image, called *perspective projection*, can also be considered as a distortion because it does not preserve angles and distances. The perspective projection of a square is a general quadrangle as demonstrated in Figure 1.2.

All the mentioned distortions are linear because they can be expressed by a linear (matrix) algebra. But there exist nonlinear distortions too. The first of them, *radial distortion*, displaces points in the image plane inwards to or

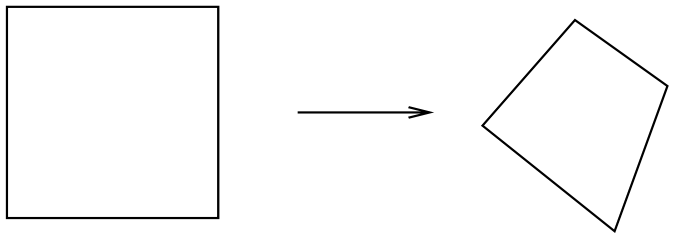


Figure 1.2: Perspective projection of a square

outwards from its optical center. The displacement is a function of a distance from the center and it is circular symmetric. If the points are shifted towards the center, the distortion is called *barrel*, otherwise *pincushion*. These names were chosen after the shapes to which a square is transformed by influence of this distortion, see Figure 1.3.

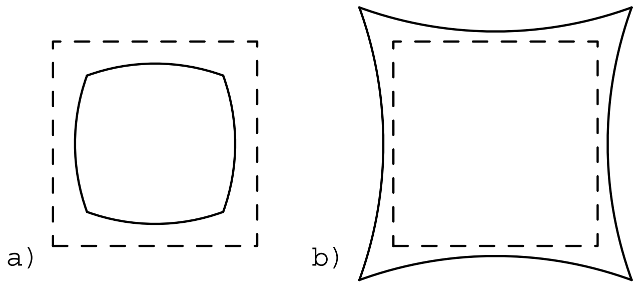


Figure 1.3: Radial distortion: a) barrel, b) pincushion

Another group of nonlinear distortions, called *lens distortions*, generally have two components, radial and tangential, as depicted in Figure 1.4. The most significant lens distortion, called *decentring*, is due to the fact that the optical centers of lens elements in composed lenses are not strictly collinear. Another one, *thin prism distortion*, arises from imperfect camera assembly.

Besides these *geometrical distortions* there exist so called *radiometrical distortions* which characterize various luminous degradations during the acquisition of images. An overview of all the distortions caused by a real camera is given in Figure 1.5.

To acquire information from an image of a scene such as shapes of objects, distances between them, color characteristics of the scene and so on, the *camera has to be calibrated.* A *camera calibration* [HS93] is a process of acquiring knowledge about the relation between an image of a scene taken by the camera and the scene itself, usually including the position of the camera in the scene and its internal characteristics.

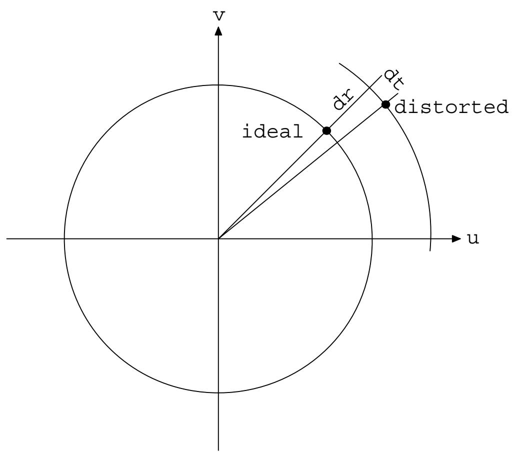


Figure 1.4: Radial (dr) and tangential (dt) components of non-linear lens distortions. An ideal point is moved into distorted position. From [WCH92]

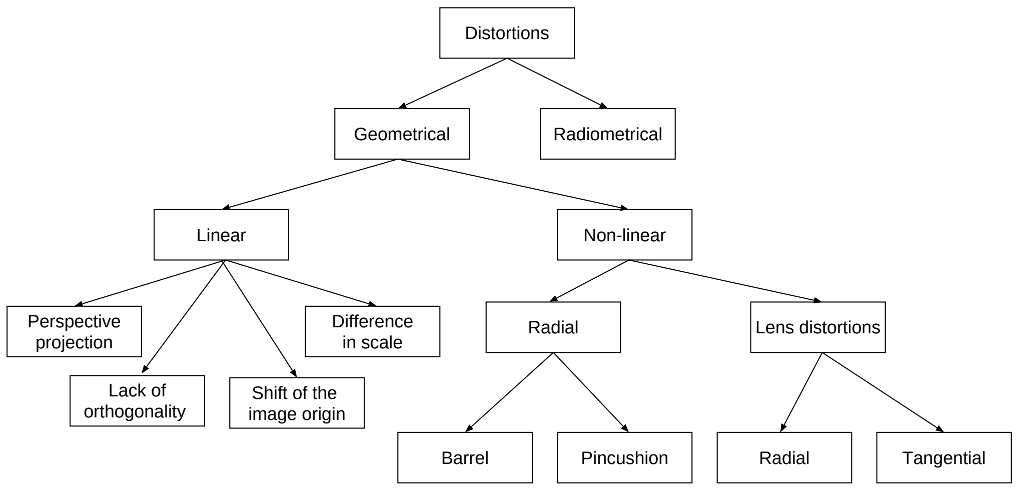


Figure 1.5: An overview of distortions caused by a real camera. Note that the radiometrical distortions are not fully covered here

The calibration consists of two stages: *geometrical* and *radiometrical*. The radiometrical calibration [KHM95], [Bey92] is performed to acquire the information on how the camera distorts the luminous characteristics of the scene such as color, luminance, brightness etc. The geometrical calibration leads to knowledge of rotation and position of the camera (commonly called the *extrinsic camera parameters*) and its internal characteristics (*intrinsic camera parameters*) such as focal length, position of the principal point, difference in scale of the image axes and so on.

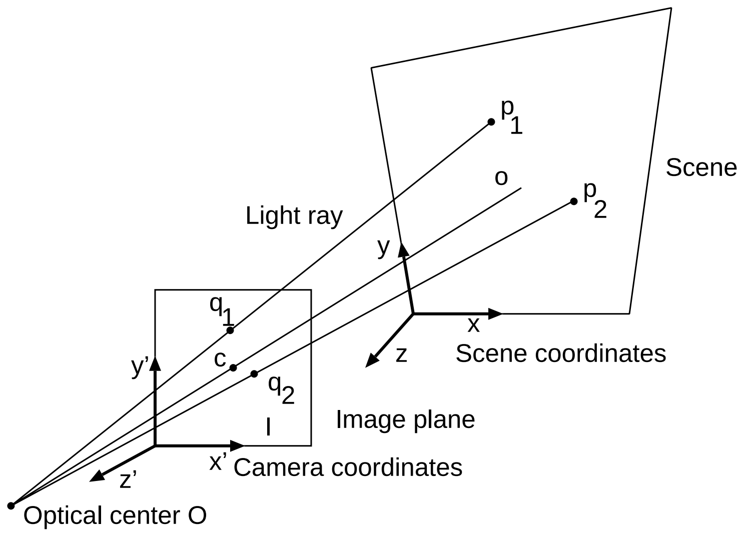


Figure 1.6: Projection of a scene into an image. See text for a description

Figure 1.6 demonstrates the projection of a scene into an image. P_i represents points in a scene and q_i are the corresponding points in the image plane. The coordinate system of the camera (x', y', z') is chosen in such a way that the z' axis is perpendicular to the image plane I. The scene coordinate system (x, y, z) can be set freely. All light rays from a scene intersect in one point, called an *optical center* or center of perspective O. The line o going through the optical center O and perpendicular to the image plane I is called an *optical axis*. The intersection between the axis o and the image plane I is called a *principal point* c. Note that this point does not have to be in the center of the CCD array.

The geometrical distortions significantly change positions of points in the image of a scene acquired by a camera. Therefore it is important to compensate such displacements. The next chapter is focused on how this is accomplished.

Chapter 2

Geometrical Calibration of a Camera

As noted before, the calibration process consists of radiometrical and geometrical stages. The radiometrical camera calibration is typically neglected in standard computer vision methods, although it plays a very important role in special applications such as astronomy imaging, photometry, color processing and so on. On the other hand, the geometrical calibration has to be performed for most of the vision tasks.

During the geometrical camera calibration, an image of some special scene is acquired. The scene contains so called *calibration objects* or *targets*, whose shapes and dimensions are a priori known. On these objects, there are visually detectable features whose coordinates are used for the calibration. It should be noted that there also exist so called *self-calibration methods* which do not require explicitly known calibration objects [ZF96], [Tri98]. Nevertheless, the known calibration target is assumed in this work.

In this Chapter, calibration objects and various types of control points are discussed first. Then, existing camera models are presented and compared.

2.1 Calibration Objects

For the camera calibration, a set of scene-image coordinate pairs has to be provided. The scene coordinates are measured directly on the calibration targets, the corresponding image coordinates has to be detected in the image. In order to allow such a detection, the targets are equipped with visually significant features painted on them. These features, such as intersections of lines, centers of gravity of circles, corners of squares and so on are then treated as *control* (*reference) points*. After detecting the points in the image and establishing the relation between their scene and image coordinates (called *matching*) the camera is ready to be calibrated.

The calibration objects can be three dimensional, virtual 3D (simulated by multiple views of a 2D target) or two dimensional (also called *coplanar*) as depicted in Figure 2.1. The 3D calibration objects are difficult to manufacture, also the measurement of the coordinates of the control points in 3D space is complicated. On the other hand, such objects allow a precise calibration. In addition, most of the calibration methods require the 3D calibration objects in order to estimate the complete set of the camera parameters.

There is a possibility to simulate a 3D calibration object by multiple views of a 2D calibration target. The 2D plane can be moved freely between the particular image acquisitions [HS97] or the movement can be constrained [Tsa87]. An usual constraint is that the plane is moved only along one direction. The main disadvantage of this approach is that the plane has to be moved precisely on a specified path and it should not be rotated or shifted during the movements. This demand can be ensured by manipulating the plane by a robot or with a help of special equipments (such as precisely made holders), both not so commonly available yet.

If a coplanar calibration object is used, the full set of camera parameters cannot be acquired without some a priori knowledge [Mel93]. For example if the focal length is not known, then the distance of the camera from the calibration object cannot be determined. In addition, when choosing initial estimates of the parameters, errors are introduced into the calibration process leading to inaccurate calibration results.

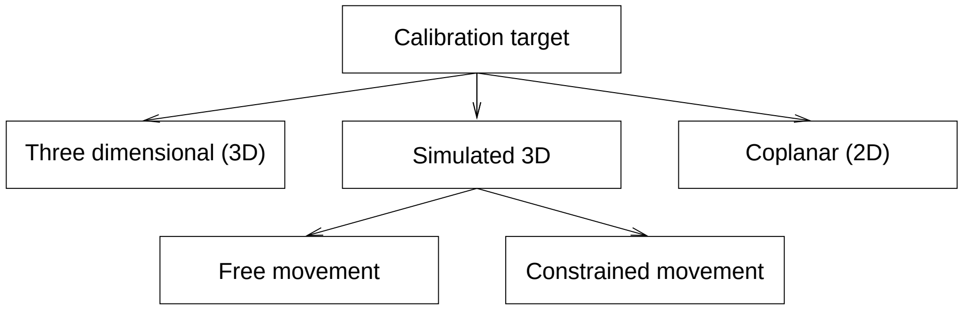


Figure 2.1: Types of calibration objects

2.2 Control Points

The control points are typically represented by visually detectable features painted on the surface of the calibration object. The points can be expressed for example as centers of gravity, intersection of lines, corners of squares or by a checkerboard as depicted in Figure 2.2.

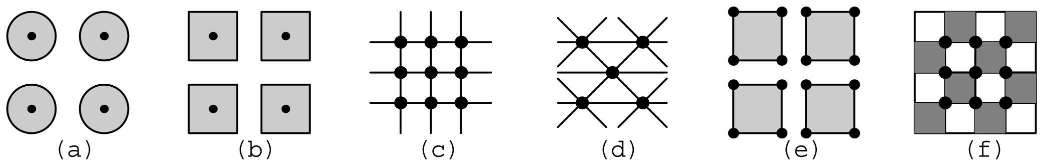


Figure 2.2: Variants of control points: centers of gravity of circles (a) and squares (b), intersections of lines in rectangular (c) and triangular (d) meshes, corners of squares (e) and checkerboard (f). The points are marked as dots

Positions of the control points in the acquired images can be detected manually or by some of the auto-detection algorithms such as [Dav97], [Shi87] or [Dev95]. The automatic detection methods typically require pure black-and-white images. Because the camera provides gray scale or color images, these images have to be converted first. The conversion, called *thresholding*, is very sensitive to the correct setting of the threshold value. Improper thresholding makes the determination of correct positions of control points in the thresholded image difficult.

For example, the line crossings method fails when the thresholded lines are too thin so they can be corrupted by noise or their parts can be invisible as depicted in Figure 2.3(a). On the other hand, when the lines are too thick and span over several pixels as demonstrated in Figure 2.3(b), their intersection cannot be properly determined.

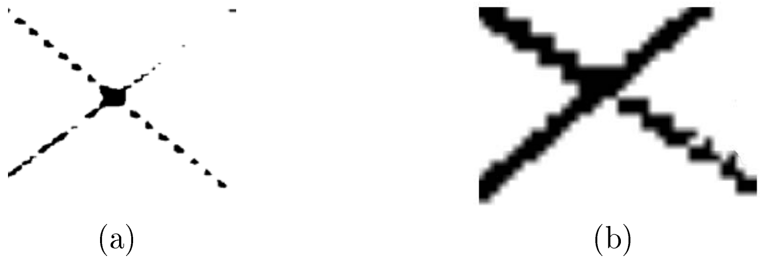


Figure 2.3: Improper thresholding of lines: (a) underthresholding causes corrupted and non-continuous lines. (b) overthresholding leads to lines spanning multiple pixels

A similar thresholding problem also applies to corners of squares. If the squares touch each other in the calibration grid (like in a checkerboard) then their thresholded images can overlap or can be separated as depicted in Figure 2.4(a). In such situations, the corners cannot be detected directly. If the squares do not form a checkerboard, their images after improper thresholding can have different sizes as demonstrated in Figure 2.4(b). The corners of such squares are then detected in wrong positions.

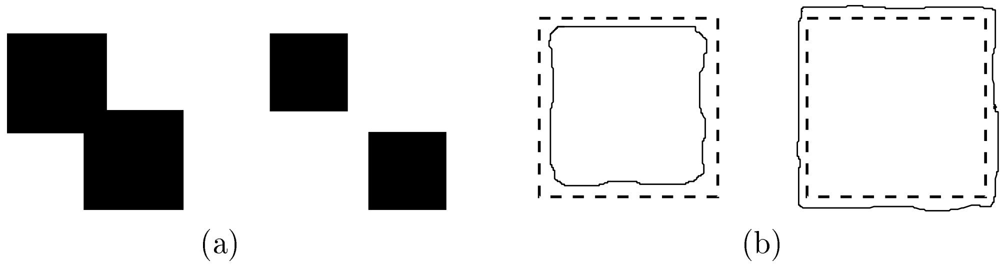


Figure 2.4: Improper thresholding of a checkerboard (a) and separated squares (b). See text for a description

A detection of centers of gravity of circular or square targets is not so sensitive to the thresholding errors but it introduces another problem. A perspective projection of a square is a general quadrangle and a circle is projected to an ellipse. In both cases, the center of gravity of the original object is not the same as the center of gravity of its perspective projection. Figure 2.5(a) depicts this situation for a square, Figure 2.5(b) for a circle. A method to solve this problem was introduced in [HS96].

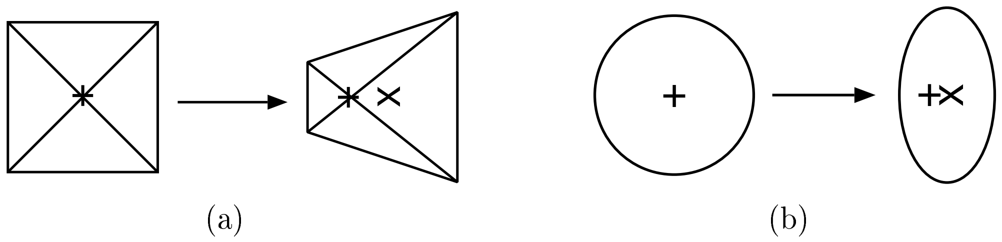


Figure 2.5: Differences between centers of gravity of an object (+) and its perspective projection (x): a) square, b) circle. See text for a description

The proper choice of the calibration target depends on the size of the calibration object, focal length of a camera, distance between the object and the camera, light conditions in the scene and many other factors. An overview of various calibration targets with their properties is summarized in Table 2.1.

Table 2.1: A comparison of various calibration targets

Target shape	Control points	Example Figure	Main advantage	Main disadvantage	Problem Figure
checkerboard	corners	2.2(f)	lower sensitivity to thresholding errors, large shape	overlapped or separated squares after thresholding	2.4(a)
squares	corners	2.2(e)	large shape allows larger distance from camera	thresholding can change position of a control points significantly	2.4(b)
squares	centers of gravity	2.2(b)	low sensitivity to thresholding errors	center of gravity of the projection is not a projection of the center of gravity of the square	2.5(a)
circles	centers of gravity	2.2(a)	low sensitivity to thresholding errors	center of gravity of the projection is not a projection of the center of gravity of the circle	2.5(b)
lines	intersections	2.2(c),(d)	can be easily detected	successful thresholding depends on thickness of lines and distance between an object and a camera	2.3

2.3 Models of a Camera

During the calibration, a real camera is simulated by a theoretical model which describes how a scene is transformed into an image. There are various *models of a camera* with different capability to cover the camera characteristics. Many of them are based on physical camera parameters but there also exist models representing only a projection of the scene points into the image. The former are called *explicit*, the latter *implicit* camera models.

An example of an explicit camera model is so called DLT model [AK71] which provides the position of a camera in a scene, its focal length, principal point and linear distortion coefficients. The extended DLT model [Mel94] also simulates nonlinear lens distortions. On the other hand, the implicit camera models such as two-plane method [WM93] do not provide any physical camera parameters at all. An overview of camera models is presented in Figure 2.6.

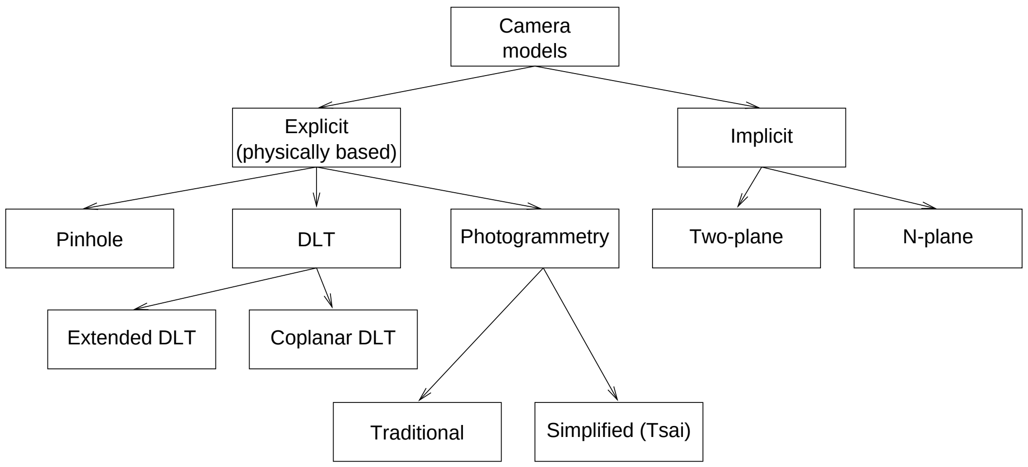


Figure 2.6: An overview of camera models

2.3.1 Pinhole Model

So called *pinhole* [Fau93] is a model of an ideal camera. The pinhole model is very simple, in fact it represents only rotation and translation (together called *rigid body transformation*) of the camera followed by a perspective projection. Another distortions (such as shift of the image origin or lens distortions) are not taken into an account. Nevertheless, the pinhole gives a good approximation of a real camera and therefore it is used as a base for other calibration methods.

The model can be formally described as

$$q_i = \mathbf{FMT}\, p_i \ , \tag{2.1}$$

where, written in homogeneous coordinates, $p_i = [x_i, y_i, z_{i,}1]^T$ are scene coordinates of the i-th point and $q_i = [w_i u_i, w_i v_i, w_i]^T$ are the appropriate coordinates in the image plane. The matrices $\mathbf{F}, \mathbf{M}, \mathbf{T}$:

$$\mathbf{F} = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \ , \tag{2.2}$$

$$\mathbf{M} = \begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{pmatrix} \ , \tag{2.3}$$

$$\mathbf{T} = \begin{pmatrix} 1 & 0 & 0 & -x_0 \\ 0 & 1 & 0 & -y_0 \\ 0 & 0 & 1 & -z_0 \end{pmatrix} \ , \tag{2.4}$$

represent focal length, rotation and translation of the camera, respectively.

The relations between the focal length f and the matrix $\mathbf{F}$ and the translation (x_0, y_0, z_0) and the matrix $\mathbf{T}$ are straightforward. On the other hand, the expression of the rotation in the matrix $\mathbf{M}$ is more complicated. The matrix is orthonormal and its columns express coordinates of the axes of the rotated coordinate system, but it has nine parameters and only three degrees of freedom.

A general rotation is typically expressed by *Euler angles* as three sequential elementary rotations ω_x, ϕ_y and κ_z along the axes of the Cartesian coordinate system. Euler angles have an unwanted singularity when $\phi_y = \frac{\pi}{2}$ or $\phi_y = \frac{3\pi}{2}$. This could be acceptable when the orientation of the camera is restricted in some directions. An alternative is to use *quaternions* (also called *Euler parameters*) [R$^+$82], which are four parameter singularity free representation of rotation. The quaternions have a descriptive geometrical meaning: if the axis of rotation is the unit vector $\mathbf{r} = [r_x, r_y, rz]^T$ and the angle of rotation along this axis is β, the rotation is represented by the quaternion

$$\mathbf{g} = \left[\cos\frac{\beta}{2}, r_x \sin\frac{\beta}{2}, r_y \sin\frac{\beta}{2}, r_z \sin\frac{\beta}{2}\right]^T \ . \tag{2.5}$$

2.3.2 Direct Linear Transform (DLT) Model

So called *direct linear transform (DLT)* [AK71] is an extension of the pinhole model. In addition to the pinhole, DLT models lack of orthogonality between image axes, shift of the image origin and difference in scale of images axes. It can be formally described as

$$q_i = \mathbf{A}\, p_i \quad , \tag{2.6}$$

where the DLT matrix $\mathbf{A}$ of the size 3×4 is composed from primitive matrices $\mathbf{V}$, $\mathbf{B}$, $\mathbf{F}$, $\mathbf{M}$, $\mathbf{T}$ as follows:

$$\mathbf{A} = \lambda \mathbf{V}^{-1}\mathbf{B}^{-1}\mathbf{FMT} \quad . \tag{2.7}$$

The factor $\lambda \neq 0$ express an overall scaling, the matrices $\mathbf{F}$, $\mathbf{M}$ and $\mathbf{T}$ represent focal length, rotation and translation as in the pinhole model (Eq. 2.2, Eq. 2.3, Eq. 2.4), the matrix $\mathbf{V}$,

$$\mathbf{V} = \begin{pmatrix} 1 & 0 & -u_0 \\ 0 & 1 & -v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.8}$$

represents the shift of the image origin and the matrix $\mathbf{B}$,

$$\mathbf{B} = \begin{pmatrix} 1+b_1 & b_2 & 0 \\ b_2 & 1-b_1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.9}$$

compensates the difference in scale and lack of orthogonality between image axes. Note that the matrix $\mathbf{V}$ is clearly invertible and the matrix $\mathbf{B}$ is invertible when $b_1^2 + b_2^2 < 1$. The values of b_1 and b_2 are commonly in the order of $10^{-4}$to 10^{-6}.

2.3.3 Coplanar DLT Model

When all object points are coplanar, that means they lie in one plane, the coplanar DLT model [Mel94] can be used. If the world coordinate system is chosen to satisfy $z_i = 0$ for all points, the third column of the DLT matrix $\mathbf{A}$ can be ignored, yielding the transformation (Eq. 2.6) in the form

$$q_i = \mathbf{A}_{(3,3)}\, p_i \quad , \tag{2.10}$$

where the coplanar DLT (CDLT) matrix $\mathbf{A}_{(3,3)}$ of the size 3×3 is defined by Eq. 2.7, except that the translation matrix $\mathbf{T}$ has to be expressed as

$$\mathbf{T} = \begin{pmatrix} 1 & 0 & -x_0 \\ 0 & 1 & -y_0 \\ 0 & 0 & -z_0 \end{pmatrix} . \tag{2.11}$$

2.3.4 Extended DLT Models

In his dissertation [Mel94], T. Melen introduced an extension to the DLT model where the nonlinear compensation term δ models both radial and decentring distortion using five more parameters. The detected image coordinates are corrected by this term after the standard DLT model is estimated. An iterative approach to determine the term δ is proposed in the paper.

J. Heikkilä and O. Silvén [HS97] use the DLT model as the initial guess for a nonlinear minimization of the following error function:

$$\sum_{i=1}^{n} (U_i - u_i)^2 + \sum_{i=1}^{n} (V_i - v_i)^2 \quad , \tag{2.12}$$

where n is the number of control points, (U_i, V_i) are the image coordinates of the points and (u_i, v_i) are the corresponding coordinates predicted by the model.

A similar approach is also applied in [LVD98]. An optimization is used to minimize the error function Eq. 2.12 with a help of multiple views of the same scene containing a 3D calibration object.

2.3.5 Traditional Photogrammetry Model

In photogrammetry, the extended pinhole model [Sla80] is used to achieve high precision results. The correction of the shift of the image origin and the compensation term $\delta = [\delta_u, \delta_v]$ are added to the original pinhole model. The term δ models both linear and nonlinear distortion, giving the transformation in the form:

$$q'_i = v + q_i + \delta(q_i) \quad , \tag{2.13}$$

where $v = [u_0, v_0]$ is the center of the image plane, q_i is a perspective projection of an object point p_i and q'_i is an appropriate point in the image plane. The accurate results are achieved at the cost of long computational time and precise calibration objects.

2.3.6 Simplified Traditional Models

Regarding the different tasks and available equipments, R. Tsai [Tsa87], [LT88] proposed a simplified traditional camera model suitable for computer vision applications. Only the radial distortion, shift of the image origin and differences in scale of the image axes are compensated by so called radial alignment constant. This camera model became very popular when its implementation in C was published freely on the Internet [Wil95]. More precise model, which also covers the decentring and thin prism distortions, was introduced by J. Weng in [WCH92].

2.3.7 Two-plane Camera Models

All above methods try to estimate the physical camera parameters, but there are situations when only the knowledge on how the camera transforms a scene into an image is needed. This does not necessary require that the physical camera parameters to be estimated at all.

Two-plane camera models define a light ray by its intersections with two parallel planes in the object coordinate system. The appropriate mapping from the image plane to these planes is estimated by a special calibration procedure based on a dense grid of control points which should cover the largest possible area of the image plane to achieve reliable results. The mapping is typically described as a combination of power terms. More than two planes and B-spline patches can also be used [CLSB92]. In addition, the parallel planes can be simulated by a static camera equipped with zoom lenses as proposed in [DLDL96].

2.3.8 Comparison of Camera Models

The choice of an appropriate camera model depends mainly on the type of application. In the field of a computer vision, the use of CCD cameras instead of precise photogrammetric equipments allows the simplified traditional or extended DLT models to be suitable for most cases. An important criterion is also the numerical stability of the particular model. When a nonlinear search is introduced during the calibration, there is a possibility that the optimization routine can stick in a local extremal point or it does not converge at all. A direct computation of the camera parameters based on a linear model (the pinhole, standard or coplanar DLT) can serve as a good initial guess. A comparison of the camera calibration models presented in previous sections is given in Table 2.2.

Table 2.2: Comparison of the camera calibration models

Camera model	Number of parameters	Advantages	Disadvantages
Pinhole	7	linear, simple, stable	does not model a real camera
DLT (Direct Linear Transform)	12	linear, stable, needs only 6 points to compute parameters	does not model non-linear distortions, needs 3D calibration object
CDLT (coplanar DLT)	11	linear, stable, needs only 6 points to compute parameters, uses 2D calibration object	does not model non-linear distortions, needs an initial guess of some of the parameters
Extended DLT	14–18	models lens distortions	iterations or non-linear search
Traditional (Photogrammetry)	14+	accurate results	non-linear search, computationally expensive, needs precise calibration object
Simplification of traditional model (Tsai)	11–16	good results, source code of the method is freely available	non-linear search, depends on the data precision
Two-plane	no physical parameters	possibility to use coplanar target, back projection can be computed	does not necessary model a real camera, accuracy strongly depends on the area of the image plane covered by the control points

Chapter 3

Camera Calibration with a Virtual 3D Calibration Object

In this Chapter, a new camera calibration method based on a virtual 3D calibration object is proposed. First, the motivation is presented. In the next Section, an overview of the proposed method is given. Then, particular steps of the method and its iterative refinement are described in more details. Finally, the proposed iterative approach is summarized and compared with other calibration methods.

3.1 Motivation

Various calibration approaches and camera models were characterized in the previous Chapter. In practice, the choice of a suitable calibration method depends mainly on the required precision of the calibration and on available resources. For the best results, the traditional photogrammetry methods have to be used. But these methods assume very fine manufactured and measured calibration objects, which are not widely available yet.

As it was already noted, a 3D calibration object is required to achieve precise and reliable results. When a coplanar object is used, the accuracy of the calibration depends on the initial guess of some of the camera parameters. If a 3D object is simulated by a multiple views of a 2D plane, it is important whether the plane can be moved freely or the movement is constrained. The latter approach requires special and usually expensive equipments such as a robot.

From the mentioned facts it follows that the most effective approach is to exploit a virtual 3D calibration object simulated by unconstrained movements of a coplanar target. Current calibration methods based on this principle either include a nonlinear search over large number of parameters [HS97] or do not provide the physical camera parameters at all [WM93]. To overcome this deficiency, the proposed method is based on a geometrical construction of the object.

3.2 Overview of the Proposed Calibration Method

The proposed method uses multiple views of a 2D calibration target (so called *calibration plate*) to simulate a 3D object, which is needed for precise and reliable calibration results. The calibration plate can be moved freely between the image acquisitions. An important assumption of the method is that the intrinsic parameters of the camera are constant for all views. This constraint allows the simulation of the 3D object to be based on relative positions of the plate between particular acquisitions.

The method consists of the following four steps (see Figure 3.1):

1. an initial estimation of the intrinsic parameters of a camera
2. an estimation of the extrinsic camera parameters for each view
3. a construction of a virtual 3D calibration object from multiple 2D planes
4. a complete camera calibration based on the virtual 3D object

Due to the use of a coplanar calibration target, an initial guess of the intrinsic camera parameters is needed. These parameters are supplied by the user in the first step of the proposed method.

In the second step, the extrinsic camera parameters are estimated for each view with respect to the provided intrinsic parameters. Any explicit coplanar camera calibration method can be applied here.

In the third step, the virtual 3D calibration object is constructed. The construction exploits the knowledge on the camera positions for all views provided by the previous step.

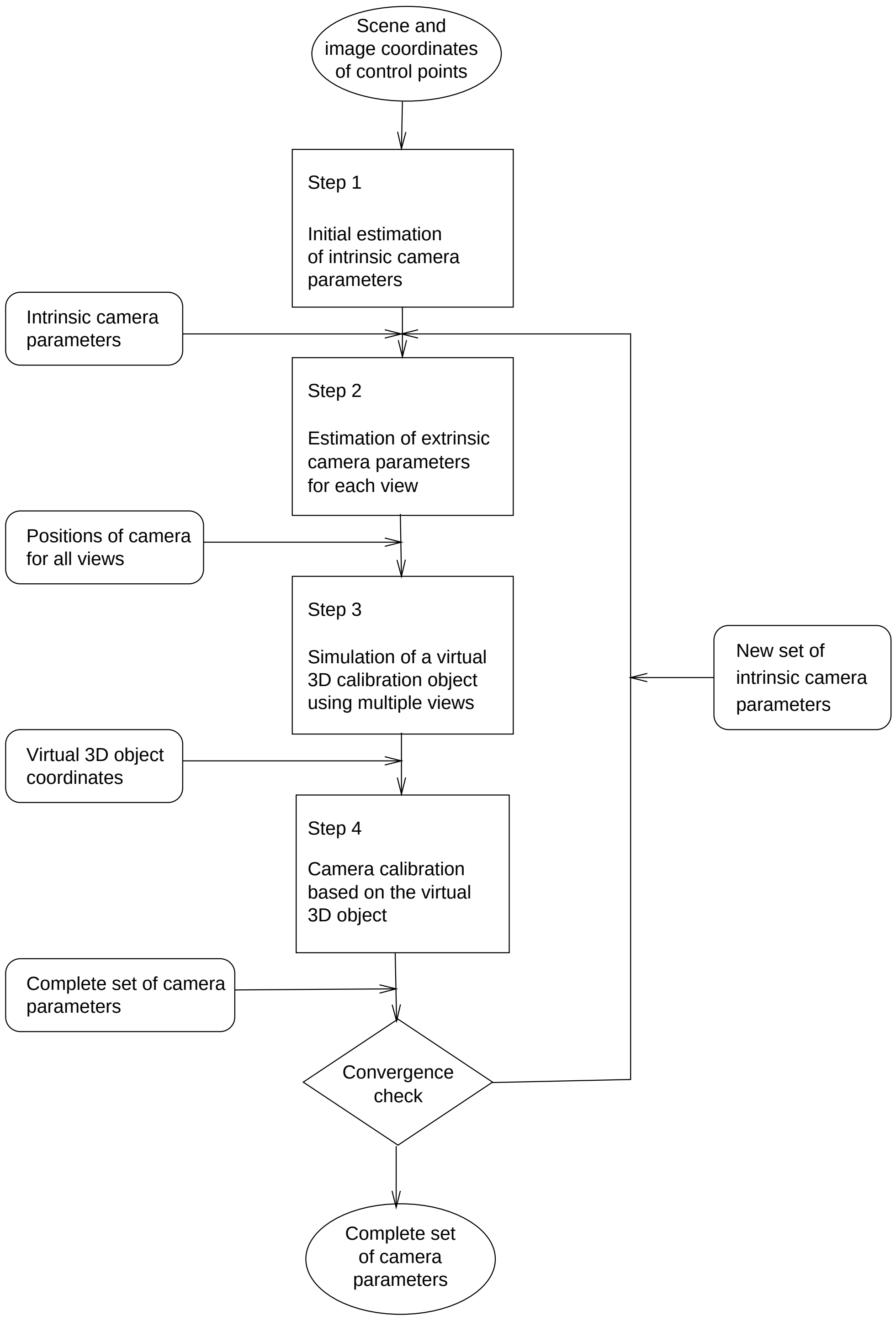

Scene and image coordinates of control points
Step 1
Initial estimation of intrinsic camera parameters
Intrinsic camera parameters
Step 2
Estimation of extrinsic camera parameters for each view
Positions of camera for all views
Step 3
Simulation of a virtual 3D calibration object using multiple views
New set of intrinsic camera parameters
Virtual 3D object coordinates
Step 4
Camera calibration based on the virtual 3D object
Complete set of camera parameters
Convergence check
Complete set of camera parameters

Figure 3.1: A scheme of the proposed calibration method

The pairs of scene and image coordinates of the control points of the simulated calibration object are finally passed to the fourth step of the proposed method, in which a complete set of the camera parameters is estimated. Any explicit 3D based camera calibration method can be used for this purpose.

As a result of the calibration, both intrinsic and extrinsic camera parameters are provided. Regarding the fact that the proposed calibration method needs an initial guess of the internal characteristics of the camera, its accuracy can be improved as follows: The obtained intrinsic parameters are put back to the second step of the method and a new set of the camera parameters is estimated. If the new estimate of the intrinsic camera parameters differ from the previous one, the whole computation is iterated. The iterations finish when some convergence check is fulfilled.

3.3 Step 1 – Initial Estimate of the Intrinsic Camera Parameters

With respect to the coplanar calibration target, an initial guess of the intrinsic camera parameters has to be is provided for the calibration. The proposed method requires a priori knowledge on the following parameters:

- focal length f
- coordinates of the principal point u_0, v_0
- linear distortion coefficients b_1 and b_2

The appropriate values can be obtained from a data sheet of the camera. If the parameters are not available, rough estimates (such as nominal focal length, center of the image and no linear distortions) should suffice. These parameters are iteratively refined during the calibration process.

3.4 Step 2 – Estimation of the Extrinsic Camera Parameters for Each View

In case of a coplanar calibration target, the estimation of the camera parameters is possible only under the knowledge of a subset of its parameters. When the intrinsic camera parameters are known, the appropriate position of the camera can be computed from the model. The intrinsic parameters are constant for all views which ensures the consistency of the estimates. The computation of this step depends on particular model of the camera, the coplanar DLT (Section 2.3.3) was chosen here. The parameters are computed directly from the model, then their values are refined in an optimization routine.

3.4.1 Direct Computation of the Extrinsic Parameters

The CDLT matrix $\mathbf{A}$ (Eq 2.7) is the result of the camera calibration with coplanar target based on the CDLT model. This matrix represents the camera and the extrinsic parameters can be extracted from it under the knowledge of the intrinsic characteristics of the camera. The matrix $\mathbf{A}$ can be written as:

$$\mathbf{A} = \lambda \mathbf{V}^{-1}\mathbf{B}^{-1}\mathbf{FMT} \ , \tag{3.1}$$

where the matrices $\mathbf{F}$ (Eq 2.2), $\mathbf{B}$ (Eq 2.9) and $\mathbf{V}$ (Eq 2.8) represent the known intrinsic camera parameters f, u_0, v_0, b_1 and b_2 respectively. The rotation matrix $\mathbf{M}$ (Eq 2.3), the translation matrix $\mathbf{T}$ (Eq. 2.11) and the scaling factor λ are unknown. They have to be extracted from the matrix $\mathbf{A}$ in order to compute their values. The matrix $\mathbf{A}$ can be premultiplied with matrices $\mathbf{F}$, $\mathbf{B}$ and $\mathbf{V}$, the result is the matrix $\mathbf{Q}$:

$$\mathbf{Q} = \mathbf{F}^{-1}\mathbf{BVA} \ . \tag{3.2}$$

From Eq. 3.1 follows that:

$$\mathbf{Q} = \lambda \mathbf{MT} \ . \tag{3.3}$$

This means that the matrix $\mathbf{Q}$ represents the unknown position of the camera and the scaling factor λ.

The extraction of the rotation matrix $\mathbf{M}$ is based on the orthogonality constraint on its columns. Eq. 3.3 can be written in a vector form as:

$$\begin{pmatrix} \vdots & \vdots & \vdots \\ \mathbf{q}_1 & \mathbf{q}_2 & \mathbf{q}_3 \\ \vdots & \vdots & \vdots \end{pmatrix} = \lambda \begin{pmatrix} \vdots & \vdots & \vdots \\ \mathbf{m}_1 & \mathbf{m}_2 & \mathbf{m}_3 \\ \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} 1 & 0 & \vdots \\ 0 & 1 & -\mathbf{t} \\ 0 & 0 & \vdots \end{pmatrix} \ , \tag{3.4}$$

yielding that the first two columns of the matrix $\mathbf{M}$ can be estimated directly from the following equations:

$$\lambda \mathbf{m}_1 = \mathbf{q}_1 \tag{3.5}$$

$$\lambda \mathbf{m}_2 = \mathbf{q}_2 \ . \tag{3.6}$$

The scale factor λ should be the same in both equations but this is not true in real situations. Regarding that, λ is approximated by the value:

$$\lambda = \frac{\|\mathbf{q}_1\| + \|\mathbf{q}_2\|}{2} \ , \tag{3.7}$$

where the operator $\|\cdot\|$ denotes the vector size of a particular vector. After the scaling factor λ is known, the first two columns of the rotation matrix $\mathbf{M}$ are approximated by the vectors:

$$\widetilde{\mathbf{m}}_1 = \frac{1}{\lambda}\mathbf{q}_1 \tag{3.8}$$

$$\widetilde{\mathbf{m}}_2 = \frac{1}{\lambda}\mathbf{q}_2 \ . \tag{3.9}$$

Being the part of the rotation matrix, these two vectors have to be orthonormal. With respect to the fact that only approximate values of the internal camera parameters and the scaling factor λ are provided, this constraint does not have to be fulfilled. Thus, the vectors are normalized first, yielding:

$$\widetilde{\widetilde{\mathbf{m}}}_1 = \frac{1}{\|\widetilde{\mathbf{m}}_1\|}\widetilde{\mathbf{m}}_1 \tag{3.10}$$

$$\widetilde{\widetilde{\mathbf{m}}}_2 = \frac{1}{\|\widetilde{\mathbf{m}}_2\|}\widetilde{\mathbf{m}}_2 \ . \tag{3.11}$$

Then, the orthogonalization is based on the following idea (see Figure 3.2): The vectors determine a plane. To become orthogonal, the vectors have to be rotated in this plane. This operation is accomplished by transforming them into the 2D coordinate system, orthogonalizing (see Figure 3.2) and transforming the orthogonalized vectors back into the 3D space. The transformation to the 2D plane is performed by a rotation matrix $\mathbf{G}$. The matrix $\mathbf{G}$ is created from a quaternion $\mathbf{g}$:

$$\mathbf{g} = \left[\cos\frac{\gamma}{2}, r_x \sin\frac{\gamma}{2}, r_y \sin\frac{\gamma}{2}, r_z \sin\frac{\gamma}{2}\right]^T = [g_0, g_1, g_2, g_3]^T \ . \tag{3.12}$$

where $\gamma = \arccos(\mathbf{n}_m \odot [0,0,1])$ is the angle of rotation along the axis $\mathbf{r} = [r_x, r_y, r_z]$. The vector $\mathbf{n}_m = \widetilde{\widetilde{\mathbf{m}}}_1 \times \widetilde{\widetilde{\mathbf{m}}}_2$ is the cross vector product of $\widetilde{\widetilde{\mathbf{m}}}_1$ and $\widetilde{\widetilde{\mathbf{m}}}_2$, $\mathbf{r} = \mathbf{n}_m \times [0,0,1]$. The operator $\odot$ denotes the scalar product of vectors. The matrix $\mathbf{G}$ is computed as [Mel94, page 31]:

$$\mathbf{G} = \frac{1}{\|\mathbf{g}\|}\begin{pmatrix} 1-2\left(g_2^2+g_3^2\right) & 2\left(g_1g_2+g_0g_3\right) & 2\left(g_1g_3-g_0g_2\right) \\ 2\left(g_1g_2-g_0g_3\right) & 1-2\left(g_1^2+g_3^2\right) & 2\left(g_2g_3+g_0g_1\right) \\ 2\left(g_1g_3+g_0g_2\right) & 2\left(g_2g_3-g_0g_1\right) & 1-2\left(g_1^2+g_2^2\right) \end{pmatrix} \ , \tag{3.13}$$

Transforming vectors $\widetilde{\widetilde{\mathbf{m}}}_1$ and $\widetilde{\widetilde{\mathbf{m}}}_2$ by the matrix $\mathbf{G}$ gives the desired coordinates of these vectors in the 2D coordinates system:

$$\begin{aligned} \mathbf{t}_1 &= \mathbf{G}\widetilde{\widetilde{\mathbf{m}}}_1 && (3.14) \\ \mathbf{t}_2 &= \mathbf{G}\widetilde{\widetilde{\mathbf{m}}}_2 && (3.15) \end{aligned}$$

Now the vectors are made orthogonal as demonstrated in Figure 3.2. The angle α between the vectors

$$\alpha = \arccos\left(\mathbf{t}_1 \odot \mathbf{t}_2\right) \qquad (3.16)$$

is changed to $\frac{\pi}{2}$ by rotating both vectors in the opposite directions by the angle $\frac{\beta}{2}$:

$$\tfrac{\beta}{2} = \tfrac{1}{2}\left(\tfrac{\pi}{2} - \alpha\right) \ . \qquad (3.17)$$

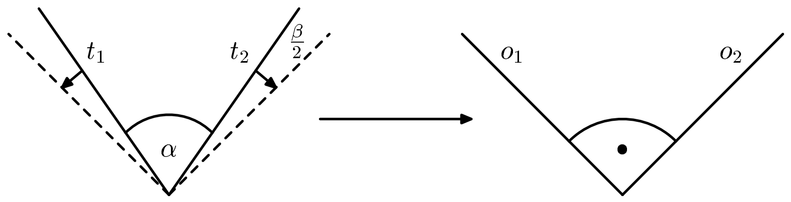


Figure 3.2: Orthogonalization of two vectors in a plane: t_1 and t_2 are the original vectors, o_1 and o_2 the orthogonalized ones

The orthogonal vectors $\mathbf{o}_1$ and $\mathbf{o}_2$ are then transformed back to the original plane by the inverse matrix $\mathbf{G}^{-1}$:

$$\begin{aligned} \mathbf{m}_1 &= \mathbf{G}^{-1}\mathbf{o}_1 && (3.18) \\ \mathbf{m}_2 &= \mathbf{G}^{-1}\mathbf{o}_2 \ . && (3.19) \end{aligned}$$

With respect to the orthogonality property of the matrix $\mathbf{M}$, its third column $\mathbf{m}_3$ is computed as the vector product of $\mathbf{m}_1$ and $\mathbf{m}_2$:

$$\mathbf{m}_3 = \mathbf{m}_1 \times \mathbf{m}_2 \ . \qquad (3.20)$$

Note that the final matrix $\mathbf{M}$,

$$\mathbf{M} = \begin{pmatrix} \vdots & \vdots & \vdots \\ \mathbf{m}_1 & \mathbf{m}_2 & \mathbf{m}_3 \\ \vdots & \vdots & \vdots \end{pmatrix} \qquad (3.21)$$

is a proper matrix of rotation.

After the rotation matrix $\mathbf{M}$ is estimated, the translation matrix $\mathbf{T}$ is computed directly from Eq. 3.4:

$$\mathbf{q}_3 = -\lambda \mathbf{M}\mathbf{t} \ , \tag{3.22}$$

yielding the vector $\mathbf{t}$ as:

$$\mathbf{t} = -\frac{1}{\lambda}\mathbf{M}^{-1}\mathbf{q}_3 \ . \tag{3.23}$$

The translation matrix $\mathbf{T}$ is then constructed as:

$$\mathbf{T} = \begin{pmatrix} 1 & 0 & \vdots \\ 0 & 1 & -\mathbf{t} \\ 0 & 0 & \vdots \end{pmatrix} \ . \tag{3.24}$$

The results of the decomposition of the CDLT matrix $\mathbf{A}$ under the knowledge of the intrinsic camera parameters are the appropriate rotation matrix $\mathbf{M}$ and the translation matrix $\mathbf{T}$, which represent the extrinsic camera parameters. In fact, these matrices encode the relative position of the camera in the particular view.

3.4.2 Finding the Optimal Extrinsic Parameters

The direct computation of the extrinsic camera parameters gives only rough estimate of rotation and translation, mainly because only approximations of the intrinsic parameters were used in Eq. 3.2. The optimal values of the extrinsic parameters have to be found so that the difference between the coordinates of the control points detected in the image and the coordinates predicted by the model is minimal. The approximated rotation and translation can be used as the initial guess for the standard least square minimization of the error function:

$$\sum_{j=1}^{n} \|q_j - \tilde{q}_j\|^2 \ , \tag{3.25}$$

where n is number of control points in the particular view, $q_j = (u_j, v_j)$ are the detected coordinates of the points and $\tilde{q}_j$ are their coordinates predicted by the model:

$$\tilde{q}_j = \mathbf{A}\, p_j \ , \tag{3.26}$$

where the matrix $\mathbf{A}$ is composed as (See Eq. 2.7):

$$\mathbf{A} = \lambda \mathbf{V}^{-1}\mathbf{B}^{-1}\mathbf{F}\mathbf{M}\mathbf{T} \ . \tag{3.27}$$

The matrices **F**, **B** and **V** represent the known intrinsic parameters (Section 2.3.3), the rotation matrix **M** is defined by three parameters σ, ρ, ϕ (Section 2.3.2) and the translation matrix **T** is determined by parameters x_0, y_0, z_0 (Eq 2.11).

The easiest way is to minimize the value of the error function Eq. 3.25 over the full set of unknowns (three for rotation – σ, ρ, ϕ – and three for translation – x_0, y_0, z_0) in $\Re^6$:

$$\min_{\sigma,\rho,\phi,x_0,y_0,z_0} \sum_{i=1}^{n} \|q_i - \tilde{q}_i\|^2 \quad . \tag{3.28}$$

But such a large number of unknowns may cause the optimization routine to fail. It would be better, if the dimension of the searching space was reduced.

3.4.3 Reduction of the Dimension of the Searching Space

The number of unknowns of a minimized function (the dimension of the searching space) can influence the speed and stability of the optimization routine. Therefore it is better to find a smallest possible number of parameters of the minimized function.

When reducing the dimension of searching space, some of the unknowns have to be made dependent of the rest of them. The question is whether to reduce the number of parameters expressing the rotation or the translation. There are constraints on the matrix of rotation, which are used in the decomposition of the CDLT matrix (see Section 3.4). On the other hand, the translation matrix is computed directly at the end of decomposition, thus it is a better candidate for the reduction of the number of parameters.

The basic idea is that, under the knowledge of the coordinates pairs of control points and their images, the x_0 and y_0 parameters of the translation can be expressed by the appropriate z_0 parameter. From the equation of perspective projection:

$$(u_j, v_j) = \left(\frac{x_j}{1 - \frac{z_j}{f}}, \frac{y_j}{1 - \frac{z_j}{f}} \right) \quad , \tag{3.29}$$

where (u_j, v_j) is a point in an image, (x_j, y_j, z_j) is its source point in a scene and f represents the focal length, it can be noticed that the coordinates of image points are function of z coordinate of their sources. When (u_j, v_j) and (x_j, y_j, z_j) are known, the equation can be reformulated. Then, for a given rotation only one unknown, z_0, to determine the translation is needed, thus reducing the dimension of the searching space to $\Re^4$ (compare with Eq. 3.28):

$$\min_{z_0,\sigma,\rho,\phi} \sum_{j=1}^{n} \|q_j - \tilde{q}_j\|^2 \quad . \tag{3.30}$$

The unknown translation has to be expressed by the matrix **A** and the coordinate pairs of control points. Denoting $\mathbf{S} = \mathbf{T}^{-1}\mathbf{A}$ and with respect to Eq. 3.27, the matrix **S** can be computed as:

$$\mathbf{S} = \mathbf{V}^{-1}\mathbf{B}^{-1}\mathbf{F}\mathbf{M} \quad . \tag{3.31}$$

The matrix **S** represents all transformations covered by the CDLT model (see Section 2.3.3), except translation. Translating the coordinates of control points p_j with the unknown translation **T** and then transforming the resulting points with matrix **S** is equal to transforming the control points with the matrix **A**:

$$q_j = \mathbf{A}p_j = \mathbf{T}\mathbf{T}^{-1}\mathbf{A}p_j = \mathbf{T}\mathbf{S}p_j \quad . \tag{3.32}$$

The result are the image coordinates of control points. Let $q' = \left[u'_j, v'_j\right]$, where:

$$\begin{pmatrix} w_j u'_j \\ w_j v'_j \\ w_j \end{pmatrix} = \mathbf{S}^{-1} \begin{pmatrix} u_j \\ v_j \\ 1 \end{pmatrix} \quad , \tag{3.33}$$

be the coordinates of the control points translated by the unknown translation and $q = [u_j, v_j]$ are coordinates of images of control points. Then the unknown translation can be expressed by z_0 reformulating Eq. 3.29 and computing the average value to minimize the error as:

$$x_0 = \frac{1}{n}\sum_{j=1}^{n} u'_j z_o - x_j \tag{3.34}$$

$$y_0 = \frac{1}{n}\sum_{j=1}^{n} v'_j z_o - y_j \tag{3.35}$$

where n is the number of control points and $p = [x_j, y_j, 0]$ is the j-th scene point. As a result of this step, the optimal translation and rotation respective to the specified values of the intrinsic parameters are estimated.

3.5 Step 3 – Construction of a Virtual 3D Calibration Object

As it was already noted, a 3D calibration object is simulated by using multiple of the same coplanar target. All views of the target are acquired by one camera in different positions, thus the internal camera parameters are assumed to be constant. One of the views is chosen as a reference view, let it be the first one.

The virtual 3D object consists of N planes, where N is the number of views. The planes are transformed so that the image coordinates of the control points observed by the camera in the reference position are the same as the image coordinates of the same points acquired by the camera in the position respective to the i-th view (see Figure 3.3). Let us denote this transformation as $\mathbf{R}_i$.

Because the positions of the camera are estimated in the previous step for all views, the transformation of the planes can be derived from them. A relationship between two positions of the camera $\mathbf{C}_{1i}$ has to be found to express the transformation of the 2D plane $\mathbf{R}_i$. The construction of the virtual 3D object can then be formally specified as:

$$q_i = \mathbf{A}_1 \mathbf{R}_i p_i \ , \ \ i = 1 \dots N \ , \tag{3.36}$$

where p_i are the scene coordinates of control points respective to the i-th view, q_i are the appropriate images, $\mathbf{A}_1$ is the matrix representing the camera in the reference position. Note that $\mathbf{C}_{11}$ and $\mathbf{R}_1$ are the identity matrices.

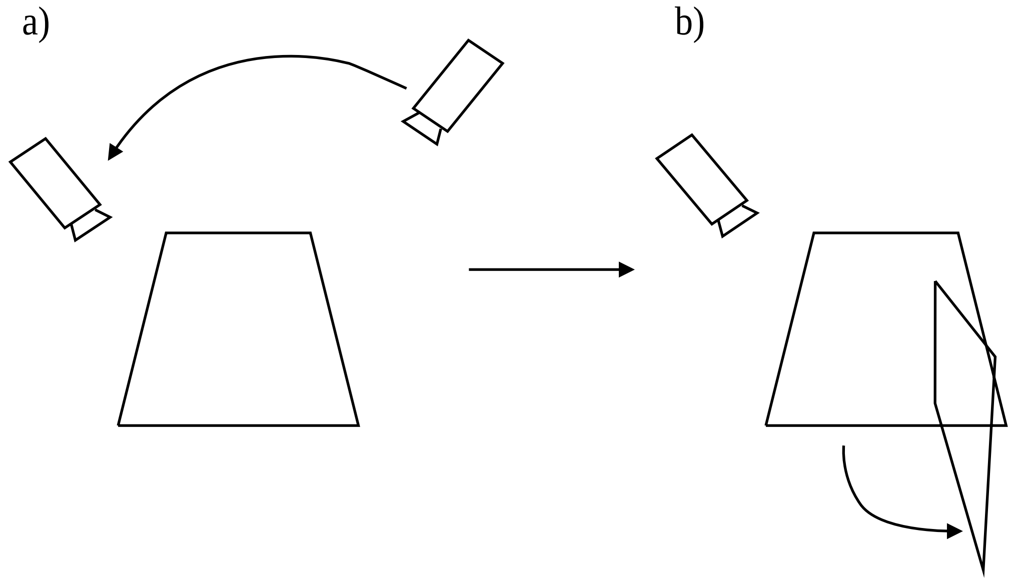


Figure 3.3: Using more views of 2D plane (a) to simulate a 3D object (b)

The relation between two positions of the camera can be expressed as:

$$\mathbf{C}_{ij} = \mathbf{T}_{ij} \mathbf{M}_{ij} \ , \tag{3.37}$$

where $\mathbf{T}_{ij}$ stands for translation and $\mathbf{M}_{ij}$ for rotation. The relation is described as follows: First, the camera has to be moved back to the center of coordinates of the scene. This is done by inverse translation. Then it has to be rotated back to the initial orientation. This is done by inverse rotation. Then the camera is moved to the the second position by rotating and translating:

$$\mathbf{C}_{ij} = \mathbf{T}_j \mathbf{M}_j \mathbf{M}_i^{-1} \mathbf{T}_i^{-1} \ , \tag{3.38}$$

where $\mathbf{T}_i$ and $\mathbf{M}_i$ are the translation and the rotation parts of the first position of the camera and $\mathbf{T}_j$ and $\mathbf{M}_j$ are the translation and the rotation parts of the second position of the camera. Homogeneous coordinates have to be used here, yielding the translation and rotation matrices in the 4×4 form instead of 3×3.

When constructing the virtual 3D object it is the 2D plane which is moved, not the camera (see Figure 3.3). The unknown position of the 2D plane in the scene coordinate system is acquired by reformulating the above expression of the relationship between the position of the camera. Utilizing homogeneous coordinates, the transformation of the 2D plane is expressed as:

$$\mathbf{R}_i = \mathbf{T}_i \mathbf{M}_i \mathbf{M}_1^{-1} \mathbf{T}_1^{-1} \ , \tag{3.39}$$

where $\mathbf{T}_1$ and $\mathbf{M}_1$ are the translation and the rotation parts of the first position of the camera and $\mathbf{T}_i$ and $\mathbf{M}_i$ are the translation and the rotation parts of the position of the camera respective to the i-th view. It can be noticed, that $\mathbf{R}_i = \mathbf{C}_{i1}^{-1}$.

Now the image coordinates of the control points acquired by the camera in the position respective to the i-th view are considered as observed by the camera in the reference position. Their appropriate scene coordinates can be found by transforming the original scene coordinates with the matrix $\mathbf{R}_i$:

$$P_i' = \mathbf{R}_i \cdot P_i \ , \tag{3.40}$$

where P_i are the original scene coordinates and P_i' are the transformed scene coordinates.

After the matrices $\mathbf{R}_i$ are found, the virtual 3D object V is constructed. The object V consists of control points. Let P_i be all the coordinates of control points respective to the i-th view. Then the virtual 3D object V can be formally specified as:

$$V = \bigcup_{i=1}^{N} \mathbf{R}_i P_i \ . \tag{3.41}$$

The appropriate images Q of the control points forming the virtual 3D object V are given as:

$$Q = \bigcup_{i=1}^{N} Q_i \ , \tag{3.42}$$

where Q_i are the coordinates of images of control points respective to the i-th view. Now there is a virtual 3D object which can be used for a complete 3D based camera calibration.

3.6 Step 4 – Calibration with the Virtual 3D Calibration Object

The virtual 3D calibration object created in the previous step is represented by control points V. The appropriate images of these points Q are also part of the output of the previous step. These coordinate pairs can be passed to any explicit camera calibration method based on a 3D calibration object. It this case, the DLT (Section 2.3.2) was chosen.

The DLT matrix $\mathbf{A}$ can be estimated using the method proposed in [Mel94, Appendix A]. Then a complete set of camera parameters is obtained by decomposition of the DLT matrix as proposed in [Mel94, Appendix C]. Thus the parameters representing the rotation, the translation, the focal length, principal points and linear distortion coefficients are obtained.

3.7 Iterative Refinement of the Camera Parameters

Provided by the complete set of the camera parameters from the previous step and with respect to the need of an initial guess, iterations are used to refine the values of the camera parameters. The intrinsic parameters obtained from the previous step are passed to the second step of the method.

At the end of each iteration, the calibration error expressing the difference between the coordinates of control points predicted by the camera model and their positions detected in the image is computed as:

$$\frac{1}{m} \sum_{i=1}^{m} \| q_i - \widetilde{q}_i \| \ , \tag{3.43}$$

where q_i are the detected points, $\widetilde{q}_i$ are the points predicted by the model (see Eq. 3.26) and m is the total number of control points in all views. If the condition of convergence is satisfied, the method is finished, else the next iteration is started. A typical conditions are for example the improvement of the camera parameters, the value of the calibration error or the maximum number of iterations.

3.8 Discussion

The proposed method is based on the coplanar calibration target. Such a target is easily accessible, but an initial guess of some of the intrinsic camera parameters is required. To overcome this problem, multiple views of the 2D target are used.

The proposed method consists of fours steps. Three of them are iterated to refine the estimated camera parameters. There is also a possibility to use nonlinear optimization of the parameters instead of iterations. The optimization can be performed either over the intrinsic parameters or a complete set of the camera parameters.

The former approach would result in a constant dimension of searching space for any number of views, it would depend on the number of intrinsic parameters. The latter approach would cause the dimension of the searching space to increase by six for each view. This method was introduced in [HS97]. It should be noted that large dimension of the searching space has bad influence on the stability of the optimization routine.

As stated before, the 2D calibration target can be easily manufactured and measured. On the other hand, the methods based on a 3D calibration object generally give more accurate results. Moreover, if the nonlinear distortion is not taken into account, a direct computation of the camera parameters is possible, which ensures stability.

The simulation of a 3D object using multiple views of a 2D plane combines the advantages of the coplanar target with the ability to estimate a complete set of the camera parameters. To avoid the stability problems, the iterative approach was chosen instead of optimization. Although there is a nonlinear minimization present in the second step of the method, the dimension of the searching space is constant for any number of views. The method of reduction of the dimension of the searching space to $\Re^4$ was proposed in Section 3.4.3. This ensures smaller number of parameters of the minimized function than in the optimization over the intrinsic parameters.

Chapter 4

Practical Realization

The proposed method is independent on camera calibration models used in the particular steps, any explicit model can be used. In our case, the DLT model and its coplanar variant were chosen. They were selected because of their simplicity and ease of implementation.

As an input, the method requires the coordinate pairs of control points consisting of the position in the scene and in the image plane. The initial guess of the focal length, principal point and linear distortion coefficients has to be supplied by the user. Such values can be acquired from the data sheet of a camera. Note that the method requires the focal length to be specified in pixels. When the conversion between the millimeters and pixels cannot be determined, a good initial guess is to multiply the focal length in millimeters by 100. The center of the CCD array is a good approximation of the unknown principal point and 0 is usually set as a value of the linear distortion coefficients. The conditions under which the iterative process stops have also be specified by the user. This includes the maximum number of iterations and desired precision level.

The proposed method is implemented in a System for Numerical Computation and Visualization – MATLAB [TMW84]. Some of the subroutines were kindly provided by Radim Halíř (estimation of DLT and CDLT matrices, linear algebra and more), other were written by the author (main method, construction of virtual 3D object, decomposition of CDLT matrix, utility functions).

MATLAB was chosen because it is widely used software in scientific community and because of the efficiency of coding. The MATLAB has a high level language with many useful build-in functions. For example the method of R. Tsai [Wil95] implemented in C has 5886 lines of code plus the optimization routine consisting of 3753 lines of code. The proposed method has 1624 lines of code including the utility functions.

The minimization of the error function defined in Eq. 3.25 requires some optimization routine to be used. In our case, SolvOpt [KK97], a freeware nonlinear optimization toolbox for the MATLAB, was chosen. The commercial optimization toolbox was not available for us, although the routines included in it can provide better and faster convergence.

Chapter 5

Experimental Results

The proposed method has been evaluated in many experiments. A set of tests with both synthetic and real data was performed and results were compared with other calibration methods widely used today.

5.1 Evaluation on Synthetic Data

To verify the reliability of the method, a series of tests with synthetic data was performed. An ideal camera with the intrinsic parameters listed in the first row of Table 5.1 was assumed in the experiments. Lens and other nonlinear distortions were not taken into account. The image size was 704 by 573 pixels and the size of the CCD cell (needed only for the conversion of the value of the focal length from millimeters to pixels), was set to $11 \times 11\mu$m. The camera was described by the CDLT model (Section 2.3.3).

As a calibration target, a regular grid of 5 by 6 points was used. Multiple views of the grid were simulated by the following imaging equations (see Eq. 2.10):

$$q_{ij} = \mathbf{A}_i \, p_j \qquad \text{for } i = 1 \ldots 6 \text{ and } j = 1 \ldots 30 \ , \tag{5.1}$$

where $p_j = [x_j, y_j, 1]$ are the homogeneous coordinates of the j-th control point in the grid, $\mathbf{A}_i$ is the appropriate CDLT matrix for the the i-th view and $q_{ij} = [w_{ij}u_{ij}, w_{ij}v_{ij}, w_{ij}]$ are the homogeneous image coordinates of the points p_j in the i-th view. The matrices $\mathbf{A}_i$ were created from the matrices $\mathbf{F}$, $\mathbf{V}$, $\mathbf{B}$, $\mathbf{T}_i$ and $\mathbf{M}_i$ as described in Section 2.3.2:

$$\mathbf{A}_i = \mathbf{V}^{-1}\mathbf{B}^{-1}\mathbf{F} \, \mathbf{M}_i\mathbf{T}_i \ . \tag{5.2}$$

Table 5.1: Intrinsic parameters of a camera simulated in the experiments with synthetic data (the first row) and their initial estimates used in the proposed calibration method (the second row). Focal length and principal point are expressed in pixels

	Focal length	Principal point		Linear coefficients	
	f	u_0	v_0	b_1	b_2
Ideal values	1136 (12.5mm)	363	280	0.02639	-0.00002
Initial guess	1300 (14.3mm)	353	286	0	0

Note that the matrices $\mathbf{F}$, $\mathbf{V}$, $\mathbf{B}$ which represent the focal length, principal point and linear distortion coefficients are the same for all views. On the other hand, the translation and rotation matrices $\mathbf{T}_i$ and $\mathbf{M}_i$, encoding the positions of a camera for particular views, differ. As a result of the simulation, a set of pairs of the scene and corresponding image coordinates of the control points is provided.

The calibration was performed by the proposed calibration method. The initial guess of the intrinsic camera parameters is listed in the second row of Table 5.1. The maximum number of iterations was set to 7. It is a reasonable limit due to the fact that the camera parameters estimated during the iterative process do not change significantly after a few iterations, as it will be discussed later in this Chapter.

In the first experiment, the ideal data without noise were passed to the method. The results of the calibration are summarized in Table 5.2. Different number of views were used, as listed in the first column of the table. The second column of the table shows the *calibration error*. The calibration error expresses the average difference between detected and predicted coordinates of the control points in the image plane and it is computed as:

$$\frac{1}{m}\sum_{j=1}^{m} \|q_j - \tilde{q}_j\| \quad , \tag{5.3}$$

where m is the number of control points in all views, q_j are coordinates of the points detected in the images and $\tilde{q}_j = \mathbf{A}_{(3,4)}\, p_j$ are image coordinates of the points predicted by the model. The DLT matrix $\mathbf{A}_{(3,4)}$ estimated in the fourth step of the method (Section 3.6) represents the camera in the reference position. Remaining columns of the table contain the intrinsic camera parameters provided by the calibration.

The first row of the table shows the error caused by the initial guess of the camera parameters, when no computations were performed. The second row contains the results for the calibration with only two planes. Although the error is small here, the camera parameters are not estimated correctly. It was stated in [LVD98] that at least three views are needed to get reliable results.

The precision does not depend so much on number of views, instead the position of the camera in these views is important. It is optimal to cover regularly the whole space around the calibration target. The last row of the table contains the ideal values of the parameters for a comparison.

For the following experiments, the data were blurred in the image plane by Gaussian noise to evaluate the reliability of the proposed method. First, the data were blurred by a noise with standard deviation set to half a pixel, which simulates the precision of the manual detection of the control points. In the next experiments, the noise with standard deviation set to one and two pixels was added to emulate additional error sources such as vibrations, incorrectly detected pints or line jitter. Finally, data were blurred by the noise with the standard deviation set to five and ten pixels to show that the method is robust and stable. The results are listed in Tables 5.3, 5.4, 5.5, 5.6 and 5.7 respectively. All tables have the same layout as Table 5.2.

As can be noticed from the presented tables, the method gives reliable results for noise with the standard deviation up to 2 pixels When the data were blurred by the noise with standard deviation of 5 pixels, the results were still a good approximation of the camera parameters. The noise with standard deviation set to 10 pixels is an extremal case. If there is such a large error in the detection of the coordinates, the calibration cannot give reliable results. This experiment was performed to show that the proposed method is robust and the calibration error is of the same order as the noise.

The calibration error after 7 iterations is shown in Figure 5.1(a). Data were blurred with Gaussian noise with standard deviation of 0.25 pixel. As can be noticed, there is no systematic linear distortion. Note that the arrows representing the shift of the control points are scaled. The mean of the error is 0.35 pixels. Figure 5.1(b) demonstrates the dependence of the calibration error on noise. It can be seen that the dependence is linear and the values of noise and the error are of the same order.

Table 5.2: Results of the calibration, synthetic data: ideal data, no noise. See text for a description

No. of views	Error	f	u_0	v_0	b_1	b_2
Initial guess	5.5349	1300	353	286	0.00000	0.00000
2	0.2683	1095	340	304	−0.00068	0.00024
3	0.2806	1110	363	276	0.00102	−0.00030
4	0.2295	1122	357	285	0.00232	−0.00064
5	0.2208	1132	355	284	0.00179	−0.00214
6	0.2119	1112	356	278	0.00142	−0.00182
Ideal values	0.0	1136	363	280	0.02639	−0.00002

Table 5.3: Results of the calibration, synthetic data: data were blurred by Gaussian noise with standard deviation of 0.5 pixel. See text for a description

No. of views	Error	f	u_0	v_0	b_1	b_2
Initial guess	5.3416	1300	353	286	0.00000	0.00000
2	0.6433	1105	342	305	0.00150	−0.00031
3	0.6344	1111	360	276	0.00020	−0.00082
4	0.6196	1118	356	283	0.00145	−0.00111
5	0.6148	1128	355	283	0.00099	−0.00267
6	0.6190	1110	356	278	0.00067	−0.00228
Ideal values	≈0.5	1136	363	280	0.02639	−0.00002

Table 5.4: Results of the calibration, synthetic data: data were blurred by Gaussian noise with standard deviation of 1 pixel. See text for a description

No. of views	Error	f	u_0	v_0	b_1	b_2
Initial guess	6.2568	1300	353	286	0.00000	0.00000
2	1.3423	1085	344	319	−0.00398	−0.00376
3	1.3461	1087	372	277	0.00084	−0.00108
4	1.3114	1118	357	292	0.00352	−0.00023
5	1.3053	1127	353	289	0.00250	−0.00235
6	1.3005	1098	354	280	0.00260	−0.00170
Ideal values	≈1.0	1136	363	280	0.02639	−0.00002

Table 5.5: Results of the calibration, synthetic data: data were blurred Gaussian by noise with standard deviation of 2 pixels. See text for a description

No. of views	Error	f	u_0	v_0	b_1	b_2
Initial guess	5.3482	1300	353	286	0.00000	0.00000
2	2.6889	1120	326	295	0.00116	−0.00766
3	2.6025	1169	366	277	0.00268	−0.00030
4	2.5637	1183	364	264	0.00345	−0.00101
5	2.4727	1167	356	277	0.00261	−0.00138
6	2.4720	1124	355	268	0.00212	−0.00264
Ideal values	≈2.0	1136	363	280	0.02639	−0.00002

Table 5.6: Results of the calibration, synthetic data: data were blurred by Gaussian noise with standard deviation of 5 pixels. See text for a description

No. of views	Error	f	u_0	v_0	b_1	b_2
Initial guess	7.7322	1300	353	286	0.00000	0.00000
2	5.5571	1257	262	383	−0.00252	−0.01947
3	5.4407	1191	298	248	0.01144	0.00542
4	5.4857	1175	239	223	0.01090	0.00382
5	5.6355	1241	306	271	0.01469	0.01289
6	5.2863	1145	303	243	0.00116	0.01517
Ideal values	≈5.0	1136	363	280	0.02639	−0.00002

Table 5.7: Results of the calibration, synthetic data: data were blurred by Gaussian noise with standard deviation of 10 pixels. See text for a description

No. of views	Error	f	u_0	v_0	b_1	b_2
Initial guess	22.8240	1300	353	286	0.00000	0.00000
2	11.8873	2385	−239	−273	0.07359	0.19145
3	12.2211	1200	215	−318	−0.10832	0.33225
4	11.6157	1389	58	29	0.04446	0.13693
5	11.7966	1502	125	169	0.04211	0.08676
6	11.8370	1352	204	63	0.00011	0.10600
Ideal values	≈10.0	1136	363	280	0.02639	−0.00002

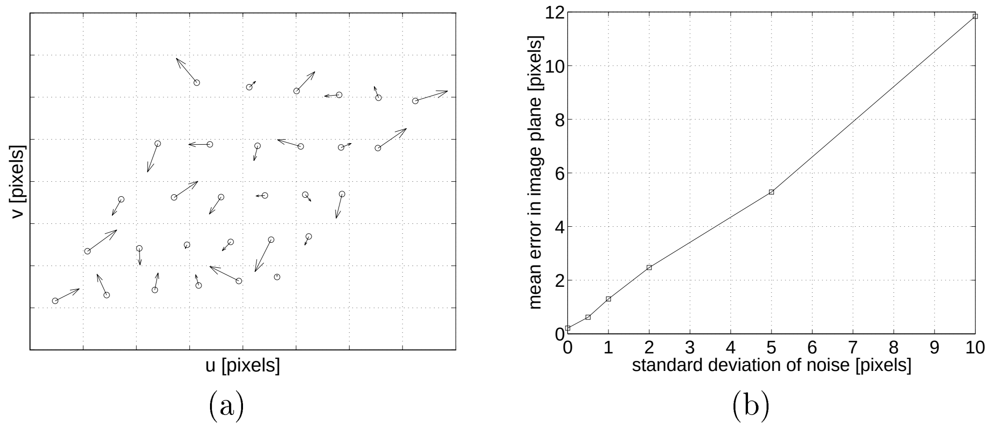


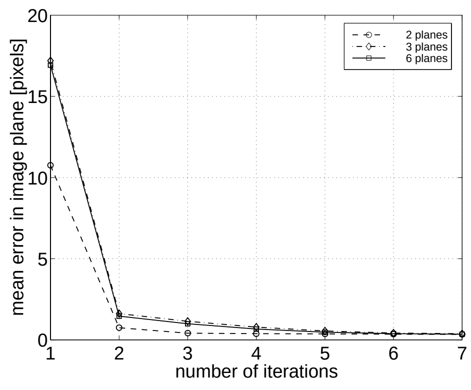
Figure 5.1: Test with synthetic data: (a) the calibration error (note that the arrows are scaled, average value is 0.35 pixels), (b) dependence of the calibration error in the image plane on noise

Figure 5.2(a) shows the development of the calibration error during an iterative process for 3, 4 and 6 planes. This test was initiated with worse guess of intrinsic camera parameters that the previous ones. After a few iteration the improvement of the error is small. The values of the parameters also do not change significantly with increasing number of iterations as can be seen in Table 5.8.

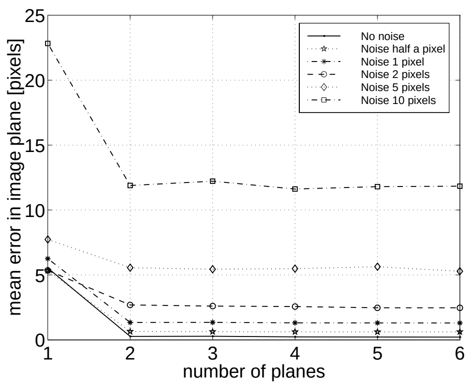
Table 5.8: Development of the calibration error and intrinsic parameters with increasing number of iterations

No. of iterations	Error	f	u_0	v_0	b_1	b_2
5	0.2295	1141	356	274	0.00211	-0.00033
10	0.2062	1132	357	279	0.00243	-0.00017
15	0.2058	1132	357	280	0.00253	-0.00001
20	0.2045	1132	357	280	0.00264	-0.00013
25	0.2035	1132	357	280	0.00274	-0.00027
30	0.2025	1132	357	280	0.00284	-0.00042
Ideal values	0	1136	363	280	0.02639	-0.00002

The dependence of the calibration error on number of planes for various noise level is depicted in Figure 5.2(b). It is clearly visible that large number of views does not necessary result in more accurate results.



(a)



(b)

Figure 5.2: Experiments with synthetic data: (a) Calibration error in image plane for 2, 3, and 6 planes. (b) Dependence of the calibration error on noise and different number of planes planes (b)

5.2 Tests with Real Data

In this part of the experiments, the calibration target was a regular grid printed on A4 paper by a laser printer. The grid consisted of 6 rows and 5 columns of lines which were 4cm in distance (see Figure 5.4(a)). To ensure the collinearity of the control points, the paper was fixed by a thin plate of glass.

Line crossings (Section 2.2) were chosen as control points because they can be easily detected and the calibration object was relatively small and in short distance off the camera. About 20 of control points were manually detected in each image with precision about half a pixel. The manual detection was performed in order to avoid additional errors caused by misdetected points.

The images, sampled to 768 x 576 pixels, were acquired by a JVC TK-S310EG camera equipped with Meopta OPTICON 1.4/16 lenses. The camera was connected to the SGI Indy workstation with a standard framegrabber. The total number of 12 images was taken under the same camera settings: the aperture set to 8 and the camera focused at the distance of 0.6 meters. The scene was illuminated from top by one light source. The experimental setup is shown in Figure 5.3.

Figure 5.4(a) shows one of the images of the scene. Enlarged area around one of the control point together with its detected position is shown in Figure 5.4(b). Note that the lines span multiple pixels and the control points is detected with sub pixel precision.

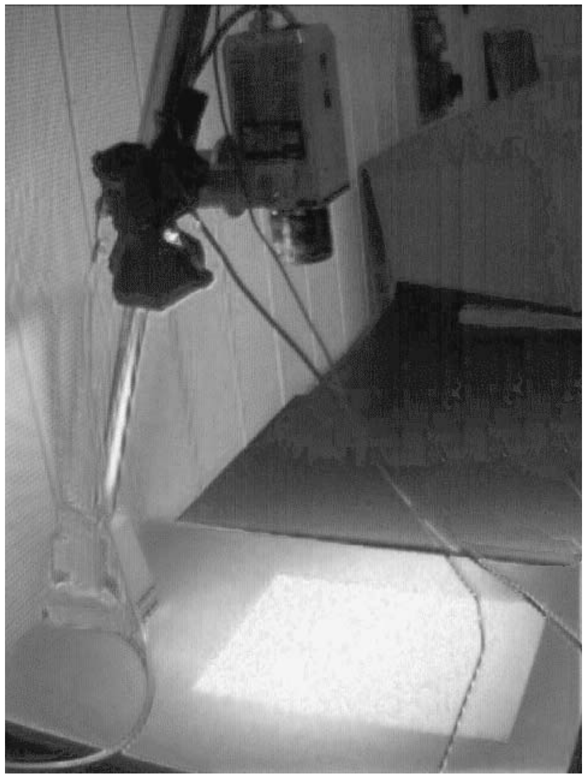
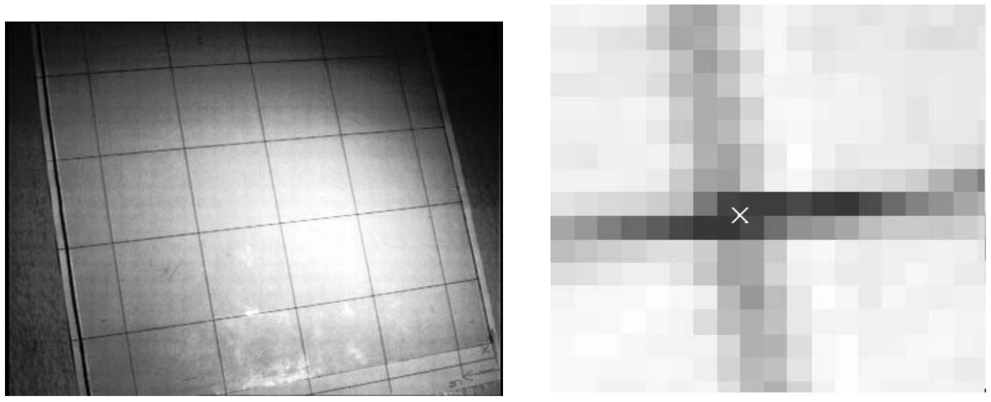
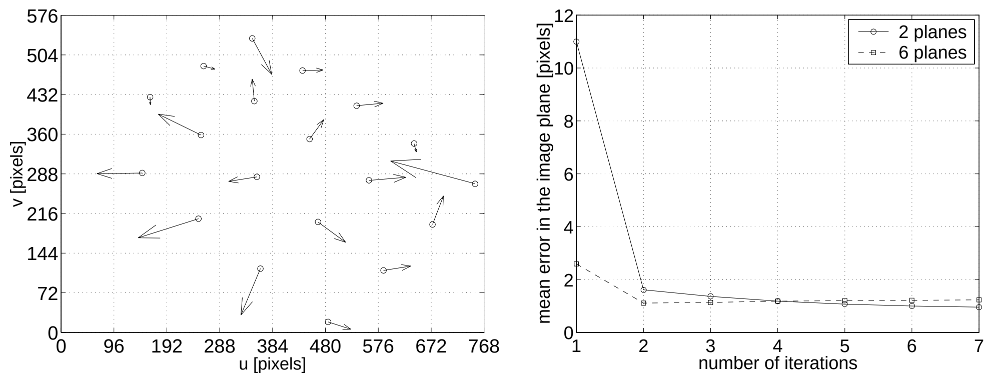


Figure 5.3: Experimental setup. The camera in the basic position is in the middle top accompanied by the light source. The calibration grid is at the bottom



(a) (b)

Figure 5.4: Example of calibration images: (a) Image taken by the camera, (b) enlarged intersection of lines with manually detected control point, marked with x



(a) (b)

Figure 5.5: Experiments on real data: (a) calibration error for the reference view (note that the arrows are scaled, average value is 0.89 pixels), (b) calibration error for different number of views (2 and 6 planes)

Figure 5.5 (a) shows the calibration error (Eq. 5.3) for the reference view. As can be noticed, there is no systematic linear distortion. The mean of the error is 0.89 pixels, note that the arrows are scaled. Possible influence of the barrel lens distortion (Chapter 1) can be observed (compare with Figure 5.1(a) where nonlinear distortions are not present). Figure 5.5(b) shows the development of the calibration error with increasing number of iterations for 2 and 6 views. In the case when 6 planes were used, the method converges after a few iterations. The insufficiency of only 2 views is also demonstrated in this figure, it can be seen that the calibration error starts to grow after the second iteration.

Table 5.9: Calibration error and intrinsic parameters for different number of views, real data. See text for a description

No. of views	Error	f	u_0	v_0	b_1	b_2
Initial guess	3.8014	1500	353	286	0	0
2	0.7188	1580	409	420	-0.00505	-0.01252
3	1.0019	1483	400	297	-0.00270	-0.00320
4	0.9718	1479	404	279	-0.00089	-0.00071
5	0.9642	1483	397	275	-0.00095	0.00045
6	0.9171	1484	397	277	-0.00120	0.00080

Table 5.9 shows the resulting values of intrinsic parameters and the calibration error (Eq. 5.3). The table has similar layout as Table 5.2, except that the ideal values could not be obtained for the real camera. According to the available informations, the value of the focal length should be around 16 mm after conversion into millimeters and the coordinates of the center of the image are 384 and 288. The approximated values of the principal point are not far from this estimate. This is common in real cameras although the principal point can be even of 60 pixels far from the center of the image plane [HS97]. The calibration error is smaller than 1 pixel. It can be considered as a good result because the control points were detected manually and nonlinear distortions were not modeled.

5.3 Comparison with Other Calibration Methods

The proposed method was compared with other freely available camera calibration methods. The most popular one is the method of R. Tsai [Tsa87], because most of the people think that it is the best method available. Another method distributed freely was developed by J. Heikkilä and O. Silvén [HS97].

The methods were compared on both synthetic and real data. Similar condition were simulated, although the Tsai's method uses only one view, while the other two methods need more views of a coplanar target. In that case, four regularly distributed views were used. For the synthetic data, the resulting intrinsic parameters and calibration error are listed in Table 5.10. Because an ideal camera was simulated, all the parameters were a priori known. The data were not blurred by any noise. There were 30 points detected in each view.

Exact values of camera parameters were not available for the test with real data. Only basic camera parameters like the nominal focal length and the camera resolution were obtained from the camera. The size of the CCD chip in millimeters needed for the conversion of the focal length from pixels to millimeters was unknown. Therefore the values of the focal length f in Table 5.11 could not be computed.

As can be noticed, the Tsai's method with a full optimization gives incorrect results, especially the principal point is estimated far away from the correct position. On the other hand, the basic variant of this method does not change the initial estimate of the principal point at all. That is why the method is unable to estimate precise results.

Table 5.10: Comparison of methods, synthetic data

Parameter name	Ideal value	Proposed method	Heikkilä and Silvén	Tsai (basic)	Tsai (full optim.)
f [pixels]	1136	1152	1147	999	1037
f [mm]	12.50	12.67	12.62	10.99	11.41
u_0 [pixels]	363	363	363	353	262
v_0 [pixels]	280	279	280	286	137
error [pixels]	0	0.23	0.0002	0.77	0.57
time [s]	—	253	24	0.09	0.9

Table 5.11: Comparison of methods, real data

Parameter name	Approx. value	Proposed method	Heikkilä and Silvén	Tsai (basic)	Tsai (full optim.)
f [pixels]	?	1513	?	?	?
f [mm]	≈16.20	?	16.64	20.32	19.55
u_0 [pixels]	≈384	400	428	353	197
v_0 [pixels]	≈288	292	314	286	330
error [pixels]	≈0.5	0.92	0.36	1.59	0.66
time [s]	—	245	11	0.06	1.75

The sensitivity to noise was tested for all the methods. This test was performed on synthetic data. The results are shown in Figure 5.6. All the methods show linear dependence of the calibration error to noise. The basic method of Tsai has larger error due to the use of only one view of a coplanar calibration target and constant value of principal point. Both the proposed method and the method of J. Heikkilä and O. Silvén give the precision of the calibration of the same order, except the ideal case with no noise, where the proposed method performs worse.

The following two tests were also performed on synthetic data. The first one evaluates the precision of the estimation of the focal length with increasing noise level. The result is depicted in Figure 5.7(a). The methods based on nonlinear optimization of all camera parameters (Tsai with full optimization, Heikkilä and Silvén) show significant variance in the estimation of the focal length. The proposed method gives the value of the focal length close to the ideal one even for bigger noise level.

The second test verifies the precision of the estimation of the principal point, again for various noise level. The results are depicted in Figure 5.7(b) as

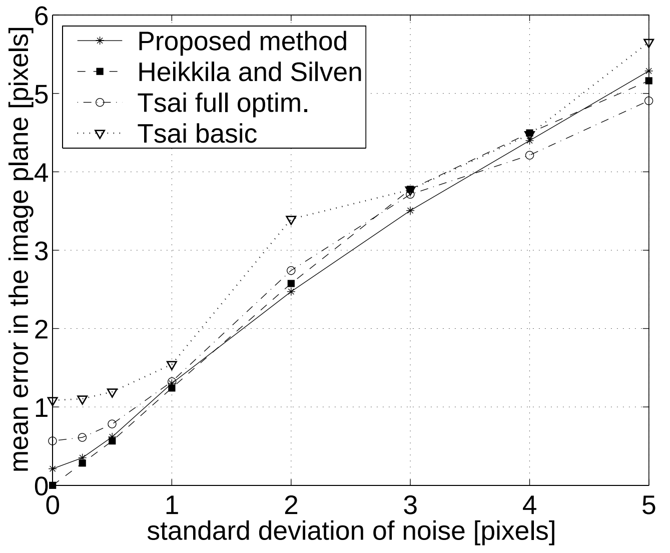
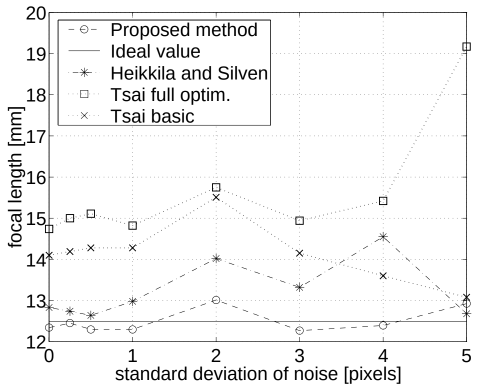


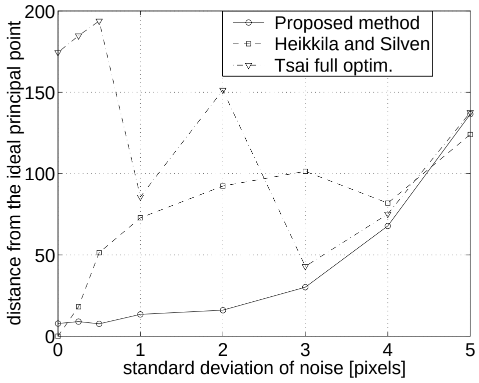
Figure 5.6: A comparison of the calibration methods. See text for a description

a distance of the estimated principal point from its ideal position. Again, the methods based on the nonlinear optimization show significant variation in the accuracy of the estimation. Especially the Tsai's method with full optimization, which was unable to estimate the correct value even when no noise was present in the data. The proposed method gives reliable results for standard deviation of noise up to 2 pixels. As can be noticed, the proposed method estimates the principal point with error which is dependent on noise. This differs from the methods based on optimization, where the error is random.

An important aspect is the computation time of the methods. The values are incomparable. Tsai's method is implemented in C and it exploits only one view. The method of J. Heikkilä and O. Silvén and the proposed method are both implemented in MATLAB. But the former uses its own minimization routine especially optimized for the task while the latter utilizes a general optimization toolkit. Without additional information such as gradients the general optimization toolkit requires large number of iteration to reach an optimal solution, which results in enormous computation time.



(a)



(b)

Figure 5.7: A comparison of the calibration methods. (a) estimation of the focal length, (b) the distance of the estimated principal point from its ideal position in pixels

5.4 Discussion

The experiments have verified that the proposed method is stable and robust against an error in the initial estimate of intrinsic parameters and to noise. More experiments have to be done with precise measured calibration object to prove the precision of the method. On the other hand, the convergence is slow when certain error level is reached.

Moreover, non-linear lens distortions are not modeled, but the method can be extended to do so. One possible way is to pass the resulting intrinsic parameters to the method of J. Heikkilä and O. Silvén [HS97] as the initial guess. Another possibility is to extend the minimization proposed in Section 3.7 with the non-linear distortion coefficients.

The main advantage of the simulation of the virtual 3D object is the constant dimension of the searching space of the nonlinear optimization, which is used in the method, for any number of views. Thanks to this fact, the method does not show random errors in the estimation of the camera parameters like the fully optimized method of R. Tsai.

Chapter 6

Conclusion and Outlook

Geometrical camera calibration is one of the basic tasks in computer vision. The character of the problem determines the requirements on the calibration method. Sometimes the high accuracy is needed, sometimes there are certain specifications of the calibration target or of the conditions under which the calibration should be performed.

The researchers are always limited in resources. This also influences the decision of what method to choose. The production of a 3D calibration object and precise measurement of coordinates of control points on this object is not always accessible to university students and small research facilities.

A 2D calibration object is a solution to the mentioned problems. But it introduces another problem, the need of an initial guess of some of the parameters of a camera. The constrained simulation of the 3D calibration object using a 2D one requires a special equipment, usually a robot. Such equipment is rarely available to small research teams.

The proposed method combines the advantages of a 2D calibration object with the extraction of the full set of the camera parameters. Although the method requires an initial guess of the intrinsic parameters of the camera, their exact values are not necessary. An iterative process refines the values of all camera parameters.

The experiments have shown that the method gives reliable results. In comparison with the most widely used method of R. Tsai, the proposed method performed better. On the other hand, it requires multiple views of a coplanar target.

Optimization routine used in the proposed method is a general one, thus needing a large number of iterations to find an optimal solution. This results in a long time of the computation. The determination of the gradients may speed this part of the method.

Another proposal for further research is to extend the method to model also the non-linear distortions. This can be done by implementing a two stage method. The first stage is the proposed method, the second one is the non-linear optimization of all the intrinsic parameters including the non-linear ones, using output from the proposed method as an initial guess.

Appendix A

Description of the Calibration Toolbox

The proposed method was implemented in the MATLAB toolbox available as a UNIX tar archive or ZIP archive for Windows platform [Bak99]. The toolbox requires version 5.0 or above of MATLAB.

The directory structure of the toolbox distribution is as follows:

- *samples* contains sample data
  - *samples/images* sample images form real data experiment
  - *samples/coords* coordinates of reference points detected in sample images
- *m* MATLAB files implementing the calibration method
- *solvopt* optimization routine used by the method's implementation

There are also several files in the root directory. The *README* file contains user instructions and the *demo.m* file contains a MATLAB function `demo`, which implements a simple demonstration of presented method.

The main function implementing the proposed method are commented. All the functions contain help information. Each file contains one function, and has the same name with the *.m* extension.

Appendix B

Toolbox User Instructions

The calibration is performed by the function `calibr8`. It has a variable number of arguments, but first three have always be passed. These are the initial estimate of the intrinsic parameters, the maximum number of iterations and the desired precision level. The first parameter, the initial estimate of intrinsic parameters, is represented by a vector of the following parameters: the focal length, the u_0 and v_0 coordinates of center of CCD chip linear distortion coefficients b_1 and b_2. The second parameter, the maximum number of iterations, is and integer number larger than zero and the third one, the precision level, is a positive floating point number.

Next parameters specify the measured data. For each plane is one variable Their number is optional. At least two planes are needed for the method to succeed. The coordinates are in form of a n by 5 matrix, where each row specifies the pairs of coordinates of control points and their images and n is a number of such a pairs in the plane. The first three columns of the matrix represent the x_i, y_i and z_i coordinate of the reference point. The fourth and the fifth column represents the u_i and v_i coordinate of the appropriate detected image of such a point.

The function returns four parameters. The first one is a vector of estimated intrinsic parameters having the same form as the initial estimate parameter of the function. Next parameters are the estimated translation and rotation matrices of the camera and errors in each plane.

The function `demo`implements simple demonstration of the proposed method. First it adds all necessary directories into the path, where MATLAB searches for functions. Then it loads the demo data located in the *samples/coords* directory and runs the calibration by invoking the `calibr8` function with the demo data. After the calibration is finished, the results are being displayed along with graphs of the resulting errors.

Appendix C

Example of the Calibration Process

A typical session with the proposed calibration toolbox is similar to this one:

At the beginning, the command:

```
>> load coordinates
```

loads the data file containing the pairs of coordinates of control points and the appropriate images. Part of a real set of coordinates is listed bellow:

40	0	0	752	270
80	0	0	674.50	196
120	0	0	585.50	112.50
160	0	0	484.80	19

In this example, the coordinates were stored in variables PC1, PC2, PC3, PC4, each variable respective to one view. The calibration was performed by invoking the method `calibr8` :

```
>> [Te, Re, Ine, errs] = calibr8( [1600, 353, 286, 0, 0], 10,
0.5, PC1, PC2, PC3, PC4 );
```

where `[1600, 353, 286, 0, 0]` is the vector containing the initial guess of the intrinsic parameters of the camera, `10` is the maximum number of iterations and `0.5` is the desired precision level. The resulting translation, rotation, intrinsic parameters estimated by the method together with the error history are stored in variables `Te` , `Re` , `Ine` and `errs` respectively.

The method displays progress information. At the beginning the calibration error caused by the initial guess of the intrinsic parameters. The the results after optimization of the extrinsic parameters, their values are displayed.

This is done for every invocation of this step of the computation, this means for all views. At the end of each iteration, the calibration error with the new intrinsic parameters and their values are printed. The example output of the method covering one iteration is listed bellow:

```
120 points, estimated DLT:
    1.5439    0.7608   -0.7187  249.9731
    0.2869    1.1767    1.5376  154.3196
    0.0009   -0.0005    0.0007    1.0000

Mean and shift for each axis:
  X: 4.101229, -0.012829
  Y: 5.906514, -0.039893
Errors in the object space: mean = 7.577898, max = 20.127003,
std = 4.543999
```

This is the calibration error for the initial guess of the intrinsic parameters. Before any extrinsic camera parameter is estimated, the following line is printed:

```
translation: x, y, z;  rotation: alpha, beta, gamma
```

For each invocation of the estimation of the extrinsic parameters, the method produces this line of output:

```
SolvOpt: Normal termination.
482.8, -429.5, 350.1;  0.748789, 0.804025, 0.002295
```

The first line is an output of the optimization routine informing that it succeeded. Then the x_0, y_0, z_0 members of the translation matrix are printed, followed by the rotation expressed by Euler angles, which are in radians.

```
SolvOpt: Normal termination.
-569.7, -429.8, 353.1;  0.745466, -0.804458, -0.002345
SolvOpt: Normal termination.
484.7, 338.1, 351.6;  -0.771088, 0.798795, 0.015045
SolvOpt: Normal termination.
-630.9, 448.7, 471.2;  -0.768865, -0.789374, -0.015623
```

At the end of the iteration, the DLT matrix estimated in the fourth step of the method is printed out together with the calibration error with respective camera parameters:

```
120 points, estimated DLT:
    1.6034    0.6006   -0.6813  246.2251
    0.2983    1.0848    1.4120  150.2781
    0.0010   -0.0007    0.0007    1.0000
Mean and shift for each axis:
  X: 0.424818, 0.000713
  Y: 0.507010, -0.002942
Errors in the object space: mean = 0.718496, max = 2.223070,
std = 0.378237
count =
    2
fl: 1213.605011, u0: 347.735054, v0: 228.109121,
b1: 0.000867, b2: 0.000241
```

The intrinsic parameters (focal length `f`, principal point `pp` and linear distortion coefficients `b`) were estimated as follows:

```
>> f = Ine(1);pp(1) = Ine(2); pp(2) = Ine(3); b(1) = Ine(4);
b(2) = Ine(5);f, pp, b

f =
   1.1319e+03
pp =
  357.0296  279.2778
b =
    0.0020   -0.0001
```

The rotation was returned in the form of the rotation matrix:

```
>> Re
Re =
    0.6991    0.4896   -0.5210
    0.0010    0.7281    0.6855
    0.7150   -0.4798    0.5085
```

To get a convenient representation of the rotation, the Euler angles (in degrees) are computed from the rotation matrix, as shown bellow:

```
>> [ox, fy, kz] = deeuler_deg( Re )
ox =
   43.3323
fy =
   45.6438
kz =
   -0.0798
```

The translation matrix was computed as:

```
>> Te
Te =
    1.0000         0  409.2888
         0    1.0000 -386.9472
         0         0  305.0442
```

Regarding to the practical realization of the method, the translation of the camera (in millimeters) is encoded in the translation matrix, and should be extracted from it by premultiplying with the rotation matrix:

```
>> Re * Te(:,3)
ans =
  -62.6208
  -72.0801
  633.3881
```

The results of the calibration are the intrinsic parameters: focal length `f = 1131`, principal point `pp = [357, 279]` and linear distortion coefficients `b = [0.0020, -0.0001]`, the rotation of the camera (represented in Euler angles): `[43.33, 45.64, -0.07]` and the translation vector (in millimeters): `[-62.62, -72.08, 633.38]`.

Bibliography

[AK71] Y. I. Abdel-Aziz and H. M. Karara. Direct linear transformation into object space coordinates in close-range photogrammetry. In *Proc. of the Symposium on Close-Range Photogrammetry, Urbana, Illinois*, pages 1–18, 1971.

[Bak99] H. Bakstein. Camera calibration toolbox, March 1999. Available by the author at *hynek@terezka.ufa.cas.cz*.

[Bey92] H. Beyer. *Geometric and Radiometric Analysis of a CCD-Camera Based Photogrammetric Close-Range System.* PhD thesis, ETH, Zürich, 1992.

[CLSB92] G. Champleboux, S. Lavallee, R. Szelinski, and L Brunie. From accurate range imaging sensor calibration to accurate model-based 3D object localization. In *Proceeding, 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 83–88, 1992.

[Dav97] E. R. Davies. *Machine Vision: Theory, Algorithms, Practicalities.* Academic Press, 2nd edition, 1997.

[Dev95] F. Devernay. A Non-Maxima Suppresion Method for Edge Detection with Sub-Pixel Accuracy. Technical Report 2724, INRIA, 1995.

[DLDL96] C. Delherm, J.M. Lavest, M. Dhome, and J.T. Lapreste. Dense reconstruction by zooming. In B. Buxton and R. Cipolla, editors, *Proc. of the 4th European Conference on Computer Vision (ECCV'96), Cambridge, England*, volume 2, pages 427–438. Springer-Verlag, 1996.

[Fau93] O. Faugeras. *Three-dimensional computer vision — A geometric viewpoint.* MIT Press, 1993.

[HS93] R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision.* Addison-Wesley, 1993.

[HS96] J. Heikkilä and O. Silvén. Calibration procedure for short focal length off-the-shelf CCD cameras. In *Proc. of the 13th International Conference on Pattern Recognition (ICPR'96), Vienna, Austria*, pages 166–170, 1996.

[HS97] J. Heikkilä and O. Silvén. A four-step camera calibration procedure with implicit image correction. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'97), San Juan, Puerto Rico*, 1997.

[KHM95] C. Kolb, P. Hanrahan, and D. Mitchell. A realistic camera model for computer graphics. In R. Cook, editor, *Proc. of the SIGGRAPH 95 Conference*, pages 317–324. ACM SIGGRAPH, Addison Wesley, August 1995.

[KK97] A. Kuntsevitch and F. Kappel. Solvopt — the solver for local nonlinear optimization problems, 1997. Available via WWW at URL: http://bedvgm.kfunigraz.ac.at:8001/alex/solvopt/index.html.

[LT88] R.K. Lenz and R.Y. Tsai. Techniques for calibration of the scale factor and image center for high accuracy 3d machine vision metrology. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 10(10):713–720, 1988.

[LVD98] J. M. Lavest, M. Viala, and M. Dhome. Do We Really Need an Accurate Calibration Pattern to Achieve a Reliable Camera Calibration? In H. Burkhardt and B. Neumann, editors, *Proc. of the 5th European Conference on Computer Vision (ECCV'98), Freiburg, Germany*, volume 1, pages 158–174. Springer-Verlag, June 1998.

[Mel93] T. Melen. Extracting physical camera parameters from the 3 by 3 direct linear transformation matrix. In A. Gruen and H. Kahmen, editors, *Optical measurements II, Zürich, Switzerland*, pages 355–365. Herbert Wichmann Verlag, Karlsruhe, Germany, 1993.

[Mel94] T. Melen. *Geometrical modelling and calibration of video cameras for underwater navigation.* PhD thesis, Institut for teknisk kybernetikk, Norges tekniske høgskole, Trondheim, November 1994.

[Nal93] V. S. Nalwa. *A guided tour of computer vision.* Addison-Wesley, 1993.

[R$^+$82] K. Rektorys et al. *A survey of applied mathematics.* SNTL, Prague, 1982.

[Shi87] Y. Shirai. *Three-Dimensional Computer Vision.* Springer-Verlag, 1987.

[Sla80] C. C. Slama. *Manual of Photogrammetry.* American Society of Photogrammetry, Falls Church, Virginia, 4th edition, 1980.

[TMW84] The MathWorks Inc. MATLAB: A system for numerical computation and visualization. *http://www.mathworks.com*, 1984.

[Tri98] B. Triggs. Autocalibration from Planar Scenes. In H. Burkhardt and B. Neumann, editors, *Proc of the 5th European Conference on Computer Vision (ECCV'98), Freiburg, Germany*, volume 1, pages 158–174. Springer-Verlag, June 1998.

[Tsa87] R. Y. Tsai. A versatile camera calibration technique for high-accuracy 3D machine metrology using off-the-shelf cameras and lenses. *IEEE Journal of Robotics and Automation*, 3(4):323–344, 1987.

[WCH92] J. Weng, P. Cohen, and M. Herniou. Camera calibration with distortion models and accuracy evaluation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14(10):965–980, October 1992.

[Wil95] R. Willson. Tsai camera calibration software, 1995. http:// www.cs.cmu.edu/afs/cs.cmu.edu/user/rgw/www/TsaiCode.html.

[WM93] G. Q. Wei and S. D. Ma. A complete two-plane camera calibration method and experimental comparsion. In *Proc. of the 4th International Conference on Computer Vision (ICCV'93), Berlin, Germany*, pages 439–446, 1993.

[ZF96] C. Zeller and O. Faugeras. Camera self-calibration from video sequences: the Kruppa equation revisited. Technical Report 2793, INRIA, 1996.